A Survey of Modern Integer Factorization Algorithms

Peter L. Montgomery

780 Las Colindas Road, San Rafael, CA 94903–2346 USA. pmontgom@cwi.nl.

Abstract

Every positive integer is expressible as a product of prime numbers, in a unique way. Although it is easy to prove that this factorization exists, it is believed very hard to factor an arbitrary integer. We survey the best known algorithms for this problem, and give some factorizations found at CWI.

AMS Subject Classification (1991): Primary 11Y05; Secondary 11A51.
CR Subject Classification (1991): F.2.1.
Keywords & Phrases: Elliptic curve method, number field sieve, primality, public–key cryptosystem, quadratic sieve.
Note: This report has not been submitted for publication elsewhere.

1. Introduction

An integer $n > 1$ is said to be a **prime number** (or simply **prime**) if the only divisors of n are ± 1 and $\pm n$. There are infinitely many prime numbers, the first four being 2, 3, 5, and 7. If $n > 1$ and n is not prime, then n is said to be **composite**. The integer 1 is neither prime nor composite.

The Fundamental Theorem of Arithmetic states that every positive integer can be expressed as a finite (perhaps empty) product of prime numbers, and that this factorization is unique except for the ordering of the factors. Table 1.1 has some sample factorizations.

$1990 = 2 \cdot 5 \cdot 199$	$1995 = 3 \cdot 5 \cdot 7 \cdot 19$	$2000 = 2^4 \cdot 5^3$	$2005 = 5 \cdot 401$
$1991 = 11 \cdot 181$	$1996 = 2^2 \cdot 499$	$2001 = 3 \cdot 23 \cdot 29$	$2006 = 2 \cdot 17 \cdot 59$
$1992 = 2^3 \cdot 3 \cdot 83$	$1997 = 1997$	$2002 = 2 \cdot 7 \cdot 11 \cdot 13$	$2007 = 3^2 \cdot 223$
$1993 = 1993$	$1998 = 2 \cdot 3^3 \cdot 37$	$2003 = 2003$	$2008 = 2^3 \cdot 251$
$1994 = 2 \cdot 997$	$1999 = 1999$	$2004 = 2^2 \cdot 3 \cdot 167$	$2009 = 7^2 \cdot 41$

Table 1.1: Sample factorizations

The existence of this factorization is an easy consequence of the definition of prime number and the well–ordering principle. The uniqueness proof is only slightly harder. However this existence proof gives no clue about how to efficiently find the factors of a large given integer. No polynomial–time algorithm for solving this problem is known.[1]

Factoring large integers has fascinated mathematicians for centuries. Gauss wrote [16, p. 398]:

[1] An algorithm is said to be **polynomial–time** if its worst case execution time is bounded by a polynomial function of the length of the input. If one wants to factor N, whose length is $\mathcal{O}(\log N)$, then an $\mathcal{O}((\log N)^{10})$ algorithm would be polynomial–time, whereas an $\mathcal{O}(N^{0.1})$ algorithm is not.

> "The problem of distinguishing prime numbers from composites, and of resolving composite numbers into their prime factors, is one of the most important and useful in all of arithmetic. The dignity of science seems to demand that every aid to the solution of such an elegant and celebrated problem be zealously cultivated."

Some books are devoted to tabulating the factors of numbers of special form. The most referenced such table is the Cunningham table [7], which lists known factors of $b^n \pm 1$ for bases $b \leq 12$ and small n. Brent et al [5] extend these tables through $b \leq 99$. Some of these factorizations appear in [32], which also has some factors of $a^n \pm b^n$. Brillhart et al [8] give factors of the Fibonacci numbers and related Lucas numbers. The RSA Challenge [34] is building a table of factorizations of partition numbers.

Factorization was once primarily of academic interest. It gained in practical importance after the introduction of the RSA public–key cryptosystem (see §5). The cryptographic strength of RSA depends upon the difficulty of factoring large numbers.

Let N be a large composite integer. Until the 1960's, the best algorithms for factoring N took time $\mathcal{O}(N^\epsilon)$ for some $\epsilon > 0$. One such algorithm is trial division, which tries to divide N by all primes up to $\sqrt{N}$. This changed when Morrison and Brillhart[27] introduced the continued fraction method, whose time [30] is

$$\exp\left\{\left(\sqrt{2}+o(1)\right)(\log N \log\log N)^{1/2}\right\} = \mathcal{O}\left(N^{\sqrt{(2+o(1))\log\log N/\log N}}\right).$$

Modern algorithms for factoring N fall into two major categories. Algorithms in the first category find small prime factors quickly. These include trial division, Pollard Rho, $P \pm 1$, and the elliptic curve method. Algorithms in the second category factor a number regardless of the sizes of its prime factors, but cost much more when applied to larger integers. These algorithms include continued fraction, quadratic sieve, and number field sieve.

In practice, algorithms in both categories are important. Given a large integer with no clue about the sizes of its prime factors, one typically tries algorithms in the first category until the cofactor (i.e., the quotient after dividing by known prime factors) of the original number is sufficiently small. Then one tries an algorithm in the second category if the cofactor is not itself prime. If one is unable to find a sufficiently small cofactor (or a prime cofactor) using methods in the first category, the factorization remains incomplete.

The interpretation of "sufficiently small" has changed considerably as technology has progressed. In the 1960's, John Brillhart and John Selfridge [9, p. 87] predicted that factoring numbers over 25 digits would be hard. In the 1970's, Richard Guy [15, p. 82] predicted that few numbers over 80 digits would be factored. In 1994 the cutoff is around 100–120 digits.

We illustrate some algorithms using $N = 1098413 = 563 \cdot 1951$. This number was selected using CWI's street address.

Williams and Shallit [40] give a computational history of factoring (and primality testing) from 1750 to 1950, i.e., before the era of electronic computers. Richard Guy [15] gives a good survey of factorization methods known in 1975. Brillhart et al [7] give a chronology of developments in factorization, both hardware and software, esp. the methods used by the Cunningham project. Robert Silverman [37] gives a more recent exposition. Several textbooks [6, 10, 16, 32] cover factorization.

We review some elementary number theory. We review some fundamental algorithms which will be needed later.

2. Notations

The symbols $\mathbb{Q}$, $\mathbb{R}$, and $\mathbb{Z}$ denote the sets of rational numbers, real numbers, and integers, respectively.

If x and y are integers, then $x \mid y$ (read: x **divides** y) means that y is a multiple of x. That is, $x \mid y$ if and only if there exists $k \in \mathbb{Z}$ such that $y = kx$.

The greatest common divisor (GCD) of two integers x and y is denoted $\gcd(x, y)$. The GCD is always positive unless $x = y = 0$. If $\gcd(x, y) = 1$, then x and y are said to be **coprime**: they have no common divisors except ± 1.

3. Review of Elementary Number Theory

This section reviews some elementary and analytic number theory. Proofs can be found in many number theory textbooks.

3.1 Congruence classes and modular arithmetic

Fix $n > 0$. Two integers x and y are said to be **congruent modulo** n if $x - y$ is divisible by n. This is written

$$x \equiv y \pmod{n}.$$

For fixed n, the $\equiv$ relation is an equivalence relation (reflexive, symmetric, transitive). The equivalence classes are called **congruence classes**. A set with exactly one representative from each congruence class is called a **complete residue system**. There are exactly n congruence classes. The **canonical complete residue system** is $\{0, 1, 2, \cdots, n-1\}$.

We omit the modulus n, writing simply $x \equiv y$, when the modulus is clear from the context.

The $\equiv$ relation is preserved under addition, subtraction, and multiplication. If x_1, x_2, y_1, y_2, and n are integers such that

$$x_1 \equiv y_1 \pmod{n} \qquad \text{and} \qquad x_2 \equiv y_2 \pmod{n},$$

then

$$\begin{aligned} x_1 + x_2 &\equiv y_1 + y_2 \pmod{n}, \\ x_1 - x_2 &\equiv y_1 - y_2 \pmod{n}, \\ x_1 x_2 &\equiv y_1 y_2 \pmod{n}. \end{aligned} \tag{3.1}$$

These are easily proved using the definition of $\equiv$. For example, $x_1x_2 - y_1y_2 = (x_1 - y_1)x_2 + y_1(x_2 - y_2)$ is a sum of two multiples of n.

Equation (3.1) says that it is meaningful to add, subtract or multiply two congruence classes, since the equivalence class of the result does not depend upon the selections of the representatives. The congruence classes modulo n form a commutative ring under these operations. This ring, denoted by $\mathbb{Z}/n\mathbb{Z}$, is called **integers modulo** n. When n is prime, division by nonzero elements is possible, and this ring is a field, often written $\mathrm{GF}(n)$.

A corollary to (3.1) is that if f is a polynomial in k variables with integer coefficients, and if $x_i \equiv y_i \pmod{n}$ for $1 \le i \le k$, then

$$f(x_1, x_2, \cdots, x_k) \equiv f(y_1, y_2, \cdots, y_k) \pmod{n}. \tag{3.2}$$

Occasionally we write $r_1 \equiv r_2$ where r_1 and r_2 are rational numbers rather than integers. The notation $a_1/b_1 \equiv a_2/b_2 \pmod{n}$ means that the numerator of $a_1/b_1 - a_2/b_2$ is divisible by n and that its denominator is coprime to n. That is, $a_1b_2 \equiv a_2b_1 \pmod{n}$ and $\gcd(b_1b_2, n) = 1$.

3.2 Properties of prime numbers

Let p be a prime number. The following properties are stated without proof:

- If $x, y \in \mathbb{Z}$, and $p \mid xy$, then $p \mid x$ or $p \mid y$. Equivalently, if $xy \equiv 0 \pmod{p}$, then $x \equiv 0 \pmod{p}$ or $y \equiv 0 \pmod{p}$.
- Unique factorization. As previously mentioned, every positive integer can be written as a (perhaps empty) product of primes, and this representation is unique except for ordering. See Table 1.1 for some examples.
- The polynomials $(X+Y)^p$ and $X^p + Y^p$ are congruent modulo p. For example, every term in

$$(X+Y)^5 - X^5 - Y^5 = 5X^4Y + 10X^3Y^2 + 10X^2Y^3 + 5XY^4$$

  is divisible by 5. This can be proved using the binomial theorem.
- (Fermat's little theorem). If $a \in \mathbb{Z}$, then $a^p \equiv a \pmod{p}$. In particular, if p does not divide a, then $a^{p-1} \equiv 1 \pmod{p}$. One proof uses the last property and induction on a.

3.3 Chinese remainder theorem

Let n_1 and n_2 be coprime positive integers. If r_1 and r_2 are arbitrary integers, then the two congruences

$$x \equiv r_1 \pmod{n_1} \qquad \text{and} \qquad x \equiv r_2 \pmod{n_2} \tag{3.3}$$

may be replaced by a single congruence

$$x \equiv r \pmod{n_1n_2}, \tag{3.4}$$

where r is chosen to satisfy $r \equiv r_1 \pmod{n_1}$ and $r \equiv r_2 \pmod{n_2}$. This is known as the **Chinese Remainder Theorem**.

For example, the two congruences

$$x \equiv 3 \pmod{13} \qquad \text{and} \qquad x \equiv 9 \pmod{17} \tag{3.5}$$

are equivalent to the single congruence $x \equiv 94 \pmod{221}$. To confirm this, note first that $1 = 52 - 51 = 4 \cdot 13 - 3 \cdot 17$; such an equation exists since 13 and 17 are coprime (cf. (4.1)). If (3.5) holds, then there exist integers k_1 and k_2 such that $x = 13k_1 + 3 = 17k_2 + 9$. Hence

$$\begin{aligned} x &= 52x - 51x = 52(17k_2 + 9) - 51(13k_1 + 3) \\ &= 884k_2 - 663k_1 + 315 = 221(4k_2 - 3k_1 + 1) + 94, \end{aligned}$$

which shows that $x \equiv 94 \pmod{221}$. Conversely, if $x \equiv 94 \pmod{221}$, say $x = 221k + 94$, then $x = 13(17k + 7) + 3 = 17(13k + 5) + 9$, implying (3.5).

A generalization allows more than two moduli. If $\gcd(n_i, n_j) = 1$ for $1 \le i < j \le k$ (i.e., if the integers $n_1, \cdots, n_k$ are pairwise coprime), and if $r_i \in \mathbb{Z}$ for all i, then the system of congruences

$$x \equiv r_i \pmod{n_i} \qquad (1 \le i \le k)$$

is equivalent to a single congruence

$$x \equiv r \pmod{n_1 n_2 \cdots n_k}, \tag{3.6}$$

when r is suitably chosen. There are efficient ways to find this r given the $\{n_i\}$ and $\{r_i\}$.

When we know an upper bound on an integer x, then we can determine x uniquely if we know it modulo enough primes. More precisely, if we know that $|x| \le B$, and if we choose $n_1 n_2 \cdots n_k \ge 2B + 1$, then (3.6) determines x uniquely.

3.4 Smooth numbers

An integer n is said to be **smooth** with respect to a bound B if no prime factor of n exceeds B. In Table 1.1, the numbers 1995, 2000, 2001, and 2002 are smooth with respect to 30.

For fixed B and $x \gg B$, the number of positive integers less than x and smooth with respect to B is approximately xu^{-u}, where $u = \ln x / \ln B$ [30, p. 94].

3.5 Density of prime numbers

The **Prime Number Theorem** states the asymptotic density of primes. Let $\pi(x)$ denote the number of primes not exceeding the real number x. Then

$$\lim_{x \to +\infty} \frac{\pi(x) \ln(x)}{x} = 1. \tag{3.7}$$

In other words, $\pi(x)$ is approximately $x / \ln(x)$ for large x.

A numerical example is $x = 1000$. Then $\pi(1000) = 168$ whereas $1000/\ln(1000) \approx 144.76$.

A more accurate approximation is $x/(\ln x - 1)$. This predicts 169.27 primes below 1000.

Riesel [32, Chapter 1] gives some history about accurate computation of $\pi(x)$ for large x.

4. Some Essential Algorithms

We will assume that the reader is familiar with the following important algorithms:

> **Multiple-precision arithmetic**[16, §4.2]. Two integers of magnitude at most N can be added, subtracted, or compared in time $\mathcal{O}(\log N)$, by operating on one digit at a time. The classical multiplication and division algorithms take time $\mathcal{O}((\log N)^2)$.

A corollary is that addition and subtraction modulo N can be done in time $\mathcal{O}(\log N)$, if the operands and result are required to be in the interval $[0,\, N-1]$. Modular multiplication can be done in time $\mathcal{O}((\log N)^2)$.

Modular exponentiation [16, pp. 441ff.]. If e is a nonnegative integer and $0 \le a < N$ then the remainder $a^e \bmod N$ can be computed with $\mathcal{O}(\log e)$ multiplications modulo N, and hence in time $\mathcal{O}((\log N)^2 \log e)$, using the binary method of exponentiation.

Greatest common divisor[16, pp. 316ff.]. If n_1 and n_2 are positive integers, then their greatest common divisor can be found in $\mathcal{O}(\log(\min(n_1,\, n_2)))$ operations on integers at most $\max(n_1,\, n_2)$. The extended GCD algorithm also finds two integers m_1, m_2 such that

$$\gcd(n_1,\, n_2) = m_1 n_1 + m_2 n_2, \qquad |m_1| \le |n_2|, \qquad |m_2| \le |n_1|. \tag{4.1}$$

5. RSA public–key cryptosystem

The RSA cryptosystem (named after its inventors Rivest, Shamir, and Adleman) [33] is the first public–key system introduced, and remains the most used public–key cryptosystem today. The strength of this cryptosystem depends upon the difficulty of factoring large integers.

A public–key cryptosystem requires each user to have his own encryption procedure E and private decryption procedure D. Everyone's E is known to all other users, like a city telephone directory. Only the user knows his D. These procedures must be bijections and inverses of each other: $D(E(M)) = E(D(M)) = M$ for all messages M. Both D and E must be cheap to execute, but it must be computationally infeasible to find D given only E.

RSA achieves these objectives by letting each user pick two large primes p and q with $p \ne q$. Let $N = pq$. Choose two exponents e, d with $de \equiv 1 \pmod{(p-1)(q-1)}$. If M is an integer, then Fermat's little theorem implies $M^{de} \equiv M \pmod{p}$ and $M^{de} \equiv M \pmod{q}$; hence $M^{de} \equiv M \pmod{N}$ for all M. Let the message space be the interval $[0,\, N-1]$ (a long message can be split into chunks). Define the encryption and decryption procedures by $E(M) \equiv M^e \pmod{N}$ and $D(M) \equiv M^d \pmod{N}$. The values of e and N are public, but d, p, q are private.

These satisfy all of the public–key requirements except possibly the requirement that it be hard to find D given E. If factoring is easy, then an intruder can find p and q given $N = pq$; from this the intruder can find $(p-1)(q-1)$ and hence d since he knows e. That is, it is easy to find D given E if factoring is easy. The converse is unknown: nobody has proven that factoring is easy if one can easily find D given E. However, the problems are believed to be equivalent.

A 1992 report [1, p. 81] recommends that any RSA public–key modulus (i.e., the product $N = pq$) be at least 1024 bits (about 309 decimal digits) if the data must remain secure for 10 years, but does not make a recommendation for longer periods.

6. Algorithms for finding small prime factors of large numbers

6.1 Trial division

If N is composite, then at least one prime divisor of N is at most $\sqrt{N}$. To factor N, the trial division algorithm successively divides N by primes 2, 3, 5, …, up to $\lfloor \sqrt{N} \rfloor$.

If p is the second largest prime factor of an integer N, then trial division takes $\mathcal{O}(p)$ steps (or $\mathcal{O}(p/\ln p)$ steps if one does trial division only by primes — see §3.5).

Almost all factoring programs attempt trial division by the smallest primes. Even if N is 1000 decimal digits long, it takes only a few seconds to divide N by all primes up to 10^7.

Sometimes the prime divisors of N are known to have a special form. For example, if $p \mid a^n - 1$ but $p \nmid a^k - 1$ for any k where $k < n$ and $k \mid n$, then $p \equiv 1 \pmod{n}$. This information facilitates trial division, since it restricts the range of possible divisors. For such numbers, one might try trial division by all qualifying primes below 2^{32} or even higher. Unless N has a special form, trial division is impractical for finding prime divisors above 10^9.

6.2 Pollard Rho

In 1974 and 1975, John Pollard announced two new algorithms for finding small factors of large integers. Each algorithm does a sequence of polynomial operations (additions, subtractions, multiplications) in such a way that intermediate results are highly composite but nonzero. Suppose N is a number to be factored and $p \mid N$. If r is one of the intermediate results and $p \mid r$, then $p \mid \gcd(r, N)$; we hope that this GCD does not equal N itself.

The Pollard Rho algorithm [29] iterates a function. Let $f \in \mathbb{Z}[X]$ be a univariate polynomial with integer coefficients. Select x_0 arbitrarily and define

$$x_{n+1} = f(x_n) \qquad (n \geq 0). \tag{6.1}$$

If p is prime, then the sequence $\{x_n \bmod p\}_{n\geq 0}$ must eventually repeat, say $x_{n_1} \equiv x_{n_2} \pmod{p}$ where $0 \leq n_1 < n_2$. Since f is a polynomial function, this and (3.2) imply $x_{n_1+k} \equiv x_{n_2+k} \pmod{p}$ for all $k \geq 0$. That is, the sequence $\{x_n \bmod p\}_{n\geq 0}$ is eventually periodic, with period dividing $n_2 - n_1$. If $n \geq n_1$ and $(n_2 - n_1) \mid n$, then $x_{2n} \equiv x_n \pmod{p}$. This is the idea behind one variation of Pollard Rho: test $\gcd(x_{2n} - x_n,\ N)$ for $n = 1, 2, \cdots$ until it is non–trivial (success) or until n is too large (failure).

$$\begin{tikzcd}
0 \arrow[r] & 1 \arrow[r] & 2 \arrow[r] & 5 \arrow[r] & 26 \arrow[r] & 114 \arrow[r] & 48 \arrow[dl] & \\
 & & & & 558 \arrow[u] & 53 \arrow[l] & & \pmod{563}
\end{tikzcd}$$

Figure 6.1: Pollard Rho cycle modulo 563 using $f(X) = X^2 + 1$ and $x_0 = 0$

Figure 6.1 shows the pattern modulo 563. The first repetition is $x_9 \equiv x_4 \equiv 26 \pmod{563}$. Since f is a polynomial with integer coefficients, the sequence has period 5 except for its first few terms: $x_{n+5} \equiv x_n \pmod{563}$ whenever $n \geq 4$. In particular, $x_{10} \equiv x_5 \pmod{563}$.

When attempting to factor $N = 1098413$ this way, the actual computations are modulo $N = 563 \cdot 1951$, but we visualize them as being done separately modulo the (unknown) prime factors of N. This abstraction is justified by the Chinese Remainder Theorem. Figure 6.2 outlines the sequence of computations (all congruences are modulo 1098413).

The cycle length is considerably longer modulo 1951, for which the first duplicate is $x_{55} \equiv x_{14} \equiv 695 \pmod{1951}$. The factor 1951 would be found while testing $\gcd(x_{82} - x_{41}, N)$, but 563 is found first, while testing $\gcd(x_{10} - x_5,\ N)$.

Complexity of Pollard Rho. By (3.2), the polynomial operations required by Pollard Rho can all be done modulo N, after which one checks $\gcd(r \bmod N,\ N)$ rather than $\gcd(r,\ N)$. This procedure ensures that the intermediate results do not grow too big, and is reflected in the data shown in Figure 6.2.

n	x_{2n-1}	x_{2n}	x_n	$\gcd(x_{2n} - x_n,\, N)$
1	$x_1 \equiv 1$	$x_2 \equiv 2$	$x_1 \equiv 1$	1
2	$x_3 \equiv 5$	$x_4 \equiv 26$	$x_2 \equiv 2$	1
3	$x_5 \equiv 677$	$x_6 \equiv 458330$	$x_3 \equiv 5$	1
4	$x_7 \equiv 394716$	$x_8 \equiv 722324$	$x_4 \equiv 26$	1
5	$x_9 \equiv 293912$	$x_{10} \equiv 671773$	$x_5 \equiv 677$	563

Figure 6.2: Factorization of 1098413 via Pollard Rho

If p is a prime dividing N, then Pollard Rho appears to take $\mathcal{O}(\sqrt{p})$ iterations to find p. Given x_{n-1} and x_{2n-2}, the n–th iteration computes

$$x_n \equiv f(x_{n-1}) \pmod{N} \qquad \text{and} \qquad x_{2n} \equiv f(f(x_{2n-2})) \pmod{N}$$

and tests $\gcd(x_{2n} - x_n,\ N)$. If $f(X) = X^2 + 1$, then this is three modular multiplications, four modular additions (subtractions are counted as additions), and one GCD operation per iteration. If $p = \mathcal{O}(\sqrt{N})$, then the time is $\mathcal{O}(\sqrt{p}(\log N)^2) = \mathcal{O}(N^{1/4}(\log N)^2)$ bit operations. This is asymptotically better than the $\mathcal{O}(p) = \mathcal{O}(N^{1/2})$ divisions needed by trial division.

It is possible to trade most of the GCDs with N for multiplications modulo N. As stated, the algorithm tests each $\gcd(x_{2n} - x_n,\ N)$. We anticipate that most GCDs will be trivial (otherwise we've found a factor). We can replace two GCD tests

$$\gcd(r,\, N) \qquad \text{and} \qquad \gcd(s,\, N) \tag{6.2}$$

by a single test

$$\gcd(rs \bmod N,\ N). \tag{6.3}$$

The GCD in (6.3) will be non–trivial if and only if at least one GCD in (6.2) is non–trivial. If rs is coprime to N, then both r and s must be coprime to N; we trade the two GCDs in (6.2) for one modular multiplication and the one GCD in (6.3). Conversely, if rs shares a factor with N, then at least one GCD in (6.2) must be non–trivial; the latter event is sufficiently rare that we can afford to test both GCDs in case they yield separate factors of N. In practice, one takes the product of several $x_{2n} - x_n$ before doing a GCD with N.

Brent [2] proposes testing $\gcd(x_m - x_n,\, N)$ whenever n is a power of 2 and $3n/2 \le m < 2n$ instead of testing values of $\gcd(x_{2n} - x_n,\, N)$. Brent's variation would find the prime 593 while testing $\gcd(x_{13} - x_8,\, N)$ rather than $\gcd(x_{10} - x_5,\, N)$, and would find the prime 1951 while testing $\gcd(x_{105} - x_{64},\, N)$ rather than $\gcd(x_{82} - x_{41}.\, N)$. Brent's variation takes about twice as many iterations, but it is about 24% faster overall because each iteration applies the polynomial f once rather than three times. Brent and Pollard [4] used this to find the 16–digit prime factor 123892 6361552897 of the 78–digit Fermat number $2^{256} + 1$.

6.3 $P - 1$

Soon after discovering Pollard Rho, John Pollard found another method which finds small prime factors. His new method, called the $P - 1$ **method** [28], finds a prime factor p of N if $p - 1$ is smooth. In the worst case, when $(p-1)/2$ is prime, the $P - 1$ method can require $\mathcal{O}(p)$ arithmetic operations to find p, as in trial division. But some prime divisors are found very quickly.

The $P-1$ method is based on Fermat's little theorem (§3.2). Suppose M is such that $(p-1) \mid M$. If $\gcd(a, N) = 1$, then $a^M \equiv 1 \pmod p$, implying $p \mid \gcd(a^M - 1, N)$. For example, if $M = 420 = 2^2 \cdot 3 \cdot 5 \cdot 7$, then this will find p if p is among

$$2,\ 3,\ 5,\ 7,\ 11,\ 13,\ 29,\ 31,\ 43,\ 61,\ 71,\ 211.$$

Eleven of the 25 primes below 100 are in this list. If we replace 420 by 11! (say), then we will also find 17, 19, 23, 37, 41, 67, 73, 89, 97, and many larger primes.

The value of M is typically the product of small primes or prime powers. If M is the product of all prime powers below a bound B, then $M \approx \exp(B)$, so the binary method of exponentiation needs $\mathcal{O}(\log M) = \mathcal{O}(B)$ multiplications modulo N to compute $a^M \bmod N$. After one GCD operation, we hope to discover p.

For example, suppose $N = 1098413$ and $B = 30$. Figure 6.3 summarizes the computations if we begin with $x_1 = a = 2$, and let x_q denote our intermediate result after processing a power of the prime q. In this example, we choose to check each $\gcd(x_q - 1, N)$ rather than only the final $\gcd(x_{29} - 1, N)$. We can stop early when we find the factor 1951 of $x_{13} - 1$.

$x_1 = 2$	
$x_2 \equiv x_1^{16} \equiv 65536$	$\gcd(x_2 - 1, 1098413) = 1$
$x_3 \equiv x_2^{27} \equiv 734876$	$\gcd(x_3 - 1, 1098413) = 1$
$x_5 \equiv x_3^{25} \equiv 639082$	$\gcd(x_5 - 1, 1098413) = 1$
$x_7 \equiv x_5^{7} \equiv 648217$	$\gcd(x_7 - 1, 1098413) = 1$
$x_{11} \equiv x_7^{11} \equiv 353244$	$\gcd(x_{11} - 1, 1098413) = 1$
$x_{13} \equiv x_{11}^{13} \equiv 304357$	$\gcd(x_{13} - 1, 1098413) = 1951$

Figure 6.3: Factoring 1098413 via $P-1$ with $x_1 = 2$ and $B_1 = 30$

The factor 1951 is found since $1951 - 1 = 2 \cdot 3 \cdot 5^2 \cdot 13$ is a product of prime powers dividing $16 \cdot 27 \cdot 25 \cdot 7 \cdot 11 \cdot 13$. On the other hand $563 - 1 = 2 \cdot 281$ is not of this form. Observe that the $P-1$ method finds 1951 first whereas trial division and Pollard Rho find 563 first. The smallest prime factor of a number is not always the easiest factor to find.

6.4 Step 2.

A modification to the $P-1$ method (called **Step 2**) allows $p-1$ to have one prime divisor exceeding B, if that prime divisor is not too big [3, 22, 26]. Specifically, suppose p is a prime divisor of N and

$$p - 1 = m \cdot q \qquad \text{where} \qquad m \mid M.$$

If $\gcd(a, N) = 1$ and $A \equiv a^M \pmod N$, then Fermat's theorem implies

$$A^q \equiv \left(a^M\right)^q = a^{Mq} = (a^{mq})^{M/m} = \left(a^{p-1}\right)^{M/m} \equiv 1^{M/m} \equiv 1 \pmod p.$$

That is, the prime p will divide $\gcd(A^q - 1, N)$.

If q is not too large, then we can find p, using the output A from Step 1. The idea [22] is to test several $\gcd(A^{n_1} - A^{n_2}, N)$ where $n_1 \neq n_2$; this will reveal p if $q \mid (n_1 - n_2)$. If we want to test all primes $q < B'$ for some B', then we can use two tables of size $\mathcal{O}(\sqrt{B'})$, one containing all values of $A^{n_1} \bmod N$ and another all values of $A^{n_2} \bmod N$. Each entry in one table is compared to each entry in the other. Variations work for the $P+1$ and elliptic curve methods, which are covered in the next two sections.

Example factors found by $P-1$ at CWI (with $B = 30$ million) are the factor p_{35} of $85^{68}+1$ and the factor p_{36} of $71^{81}+1$, where

$$\begin{aligned} p_{35} &= 11246\,3189495079\,4641128208\,4363679513, \\ p_{36} &= 296390\,4308479769\,5878152861\,5585508917, \\ p_{35}-1 &= 2^3 \cdot 7 \cdot 17^2 \cdot 11177 \cdot 327881 \cdot 628997 \cdot 1409467 \cdot 213884611, \\ p_{36}-1 &= 2^2 \cdot 3^5 \cdot 283 \cdot 739 \cdot 5347 \cdot 7699 \cdot 37589 \cdot 24474559 \cdot 38498773. \end{aligned}$$

These factors appear in the update to [5].

6.5 $P+1$

The $P-1$ method finds a factor p of N if $p-1$ is sufficiently smooth. The method has found many factors, but fails miserably if $p-1$ has a very large factor, such as if $p-1 = 2q$ for some prime q.

In 1982, Hugh Williams [39] published a method which works when $p+1$ (rather than $p-1$) is smooth. Williams's method, called the $P+1$ **method**, operates in the finite field $\mathrm{GF}(p^2)$ having p^2 elements and characteristic p.

Let P be an integer and assume that P^2-4 is is a quadratic non–residue (i.e., not a square) modulo p. Denote $f(X) = X^2 - PX + 1$. This quadratic has two roots α and α^{-1} over $\mathrm{GF}(p^2)$, which satisfy $\alpha + \alpha^{-1} = P$. Because $P^p \equiv P \pmod{p}$, one root of f is α^p:

$$f(\alpha^p) = \alpha^{2p} - P\alpha^p + 1 = (\alpha^2 - P\alpha + 1)^p = f(\alpha)^p = 0 \qquad \text{in } \mathrm{GF}(p^2).$$

Since f has only two roots, either $\alpha^p = \alpha$ or $\alpha^p = \alpha^{-1}$. If $\alpha^p = \alpha$, then $\alpha \in \mathrm{GF}(p)$, and

$$P^2 - 4 = \left(\alpha + \alpha^{-1}\right)^2 - 4 = \left(\alpha - \alpha^{-1}\right)^2.$$

This is impossible since $\alpha - \alpha^{-1} \in \mathrm{GF}(p)$ and P^2-4 is assumed to be a quadratic non–residue modulo p. Therefore $\alpha^p = \alpha^{-1}$, implying $\alpha^{p+1} = 1$.

The $P-1$ method selects a in the multiplicative group $\mathrm{GF}(p)^*$ of order $p-1$, and finds p if this order is smooth. The $P+1$ method is structurally similar to $P-1$, but takes powers of $\alpha \in \mathrm{GF}(p^2)$ and succeeds if the order of α is smooth. One way to do the arithmetic observes that α and 1 is a basis for $\mathrm{GF}(p^2)$ over $\mathrm{GF}(p)$, and uses arithmetic modulo N in place of arithmetic modulo the (unknown) prime p. This can be improved considerably by using Lucas functions to manipulate values of $\alpha^e + \alpha^{-e}$ rather than α^e [39].

There is no known way to check beforehand whether P^2-4 is a non–residue without knowing p. If one runs the method three times using three values for P, then there is an 87.5% chance that at least one of the values for P^2-4 will be a quadratic non–residue. When P^2-4 is a quadratic residue, then $\alpha \in \mathrm{GF}(p)$, and the $P+1$ method becomes an expensive variant of the $P-1$ method.

One factor found by $P+1$ at CWI (with $B = 30$ million) is the factor p_{37} of $45^{123}+1$, where

$$\begin{aligned} p_{37} &= 4190453\,1519402086\,5671558238\,2315221647, \\ p_{37}+1 &= 2^4 \cdot 283 \cdot 2423 \cdot 21881 \cdot 39839 \cdot 1414261 \cdot 2337233 \cdot 132554351. \end{aligned}$$

6.6 Elliptic Curve Method

The $P \pm 1$ methods find a factor p of N if either $p \pm 1$ is sufficiently smooth. However they fail if both $p-1$ and $p+1$ have large prime factors. This happens frequently; for example, both $(p-1)/4$ and $(p+1)/6$ are primes for $p = 29$, 173, 317, 653, 893. In 1985, Hendrik Lenstra, Jr. [21] overcame this difficulty when he announced a similar method called the **Elliptic Curve Method**, abbreviated ECM.

An **elliptic curve** over a field K of characteristic not 2 is the set of solutions $(x, y) \in K \times K$ to a cubic equation

$$Y^2 = X^3 + AX^2 + BX + C, \tag{6.4}$$

together with a special point (conceptually (∞, ∞)) called the **point at infinity**. There is one restriction to the coefficients in (6.4): the discriminant of the cubic polynomial must be nonzero, i.e.,

$$-4A^3C + A^2B^2 + 18ABC - 4B^3 - 27C^2 \neq 0.$$

The points on an elliptic curve form an abelian group $E(K)$ when the group operations are suitably defined, as illustrated in Figure 6.4. The negation of the point at infinity is itself; the negation of any other point $P_1 = (x_1, y_1)$ is defined to be $-P_1 = (x_1, -y_1)$. For addition, suppose that P_1 and P_2 are two points on the elliptic curve. If either P_1 or P_2 is the point at infinity, then define $P_1 + P_2$ to be the other point. Otherwise suppose $P_1 = (x_1, y_1)$ and $P_2 = (x_2, y_2)$. If $x_1 \neq x_2$, then define $P_1 + P_2 = -P_3$, where P_3 is the point where the straight line through P_1 and P_2 re–intersects (6.4). A calculation gives

$$P_3 = (x_3, y_3), \qquad \text{where} \qquad \begin{aligned} m &= \frac{y_2 - y_1}{x_2 - x_1}, \\ x_3 &= m^2 - A - x_1 - x_2, \\ y_3 &= y_1 + m(x_3 - x_1). \end{aligned} \tag{6.5}$$

When instead $x_1 = x_2$, then $y_1^2 = y_2^2$ by (6.4). If $y_1 = -y_2$, then define $P_1 + P_2$ to be the point at infinity. If $y_1 \neq y_2$ (so that $y_1 = -y_2 \neq 0$), then define $P_1 + P_2 = -P_3$, with P_3 defined as in (6.5), except that we use $m = (3x_1^2 + 2Ax_1 + B)/2y_1$ (slope of tangent line at $P_1 = P_2$).

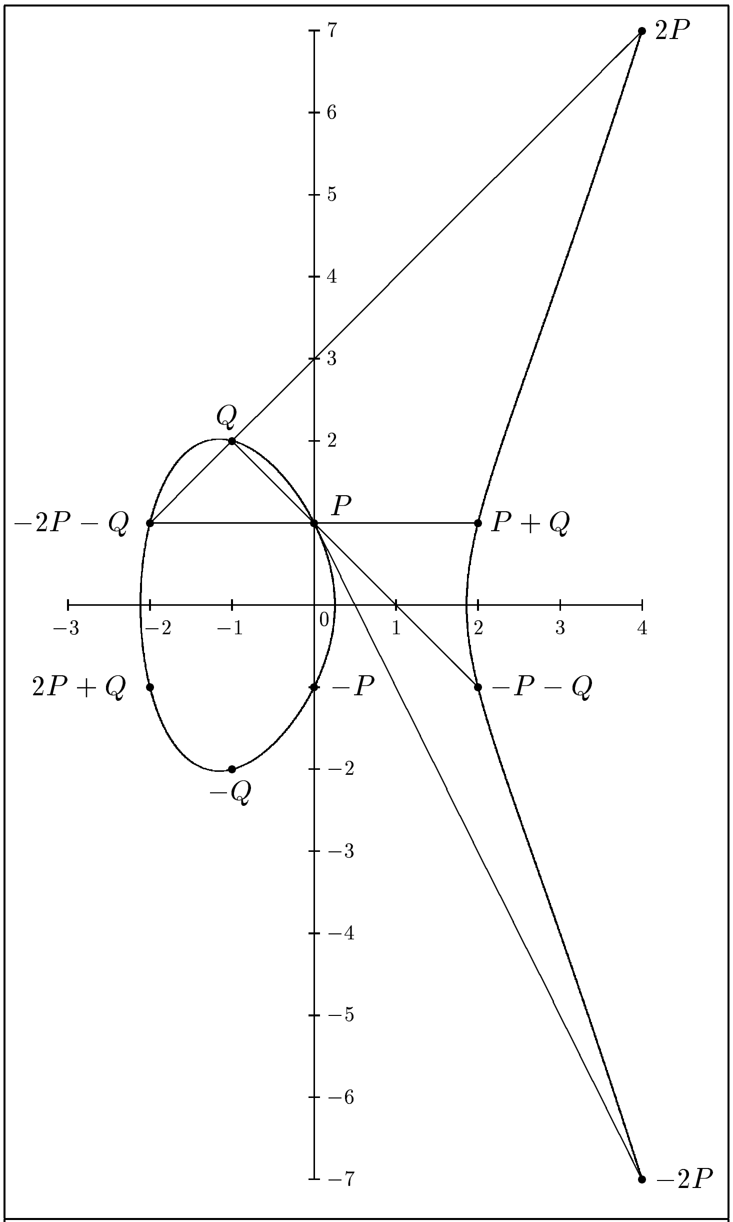
It is amazing that this $+$ defines an associative operation. All group operations are defined in terms of ordinary addition, subtraction, multiplication, division, and comparison (no square roots), and are meaningful over arbitrary fields where $2 \neq 0$. In particular, they are meaningful if $K = \mathrm{GF}(p)$, where p is an odd prime. The resulting elliptic curve group $E(\mathrm{GF}(p))$ is finite; Hasse [35, p. 131] showed that its order is $p + 1 - \tau$ where $|\tau| \leq 2\sqrt{p}$. By changing the constants in (6.4), we get another curve, whose order is usually different.

If N is composite, say $N = pq$ where p and q are distinct odd primes, then the ring $\mathbb{Z}/N\mathbb{Z}$ is not a field, but we can use (6.4) to define a curve and attempt to use the algebraic rules to do group operations modulo N. We will fail (i.e., be unable to execute the algebraic operations) only if we attempt to divide by a nonzero, non–invertible number modulo N [21]. Such a denominator (called a **zero divisor**) will be divisible by p or q but not both, and will give us a factor of N.

For example, suppose $N = 1098413$. We might choose

$$E : Y^2 = X^3 - X + 1 \qquad \text{and} \qquad P_0 = (0, 1).$$

It turns out that $|E(\mathrm{GF}(563))| = 560 = 2^4 \cdot 5 \cdot 7$ and $|E(\mathrm{GF}(1951))| = 1948 = 2^2 \cdot 487$. If we attempt to compute $560P_0$ (or any multiple thereof), then we will strike the identity element of the group modulo 563, but probably not modulo 1951. This will cause us to attempt to divide by a nonzero multiple of 563, and the factor 563 will be found. Indeed, if we work modulo 1098413, then



$$
\begin{aligned}
2P &= (0,\,1) + (0,\,1) = (4,\,7)\\
P+Q &= (0,\,1) + (-1,\,2) = (2,\,1)\\
2P+Q &= (4,\,7) + (-1,\,2) = (-2,\,-1)\\
P+(P+Q) &= (0,\,1) + (2,\,1) = (-2,\,-1)
\end{aligned}
$$

Figure 6.4: Group law on $y^2 = x^3 - 4x + 1$

$$\begin{aligned}
P_0 &= (0,\ 1),\\
2P_0 &= (823810,\ 411904),\\
4P_0 &= (351660,\ 356515),\\
8P_0 &= (1009192,\ 539351),\\
16P_0 &= (1097285,\ 905229),\\
32P_0 &= (258049,\ 365818),\\
64P_0 &= (759179,\ 793734),\\
80P_0 = 64P_0 + 16P_0 &= (590036,\ 204995),\\
160P_0 &= (196136,\ 560546),\\
320P_0 &= (252057,\ 444662),\\
640P_0 &= (289957,\ 901426).
\end{aligned}$$

Next we attempt to compute

$$560P_0 = 640P_0 - 80P_0 = (289957,\ 901426) + (590036,\ -204995).$$

The x–coordinates 289957 and 590036 are distinct, but their difference

$$590036 - 289957 = 300079 = 13 \cdot 41 \cdot 563$$

is not invertible modulo $N = 563 \cdot 1951$. A GCD finds the factor 563.

One early number done by ECM is the 843–rd Fibonacci number F_{843}. Its algebraic cofactor has five prime factors, namely

$$\frac{F_{843}}{2F_{281}} = p_{12} \cdot p_{13} \cdot p_{15} \cdot p_{16} \cdot p_{63},$$

where

$$\begin{aligned}
p_{12} &= 46\,6269593837, & p_{12} - 1 &= 2^2 \cdot 3067 \cdot 38006977,\\
& & p_{12} + 1 &= 2 \cdot 3 \cdot 41 \cdot 71 \cdot 281 \cdot 95003,\\
p_{13} &= 257\,6582465657, & p_{13} - 1 &= 2^3 \cdot 10957 \cdot 29394251,\\
& & p_{13} + 1 &= 2 \cdot 3 \cdot 281 \cdot 1279 \cdot 1194857,\\
p_{15} &= 81830\,3948755277, & p_{15} - 1 &= 2^2 \cdot 20457\,5987188819,\\
& & p_{15} + 1 &= 2 \cdot 3 \cdot 53 \cdot 281 \cdot 757 \cdot 12097213,\\
p_{16} &= 238537\,7797192381, & p_{16} - 1 &= 2^2 \cdot 3^3 \cdot 5 \cdot 281 \cdot 3191 \cdot 4926407,\\
& & p_{16} + 1 &= 2 \cdot 193 \cdot 1579 \cdot 7841 \cdot 499133,
\end{aligned}$$

and p_{63} is a 63–digit prime. The $P - 1$ method (with $B = 20000$ and a Step 2 bound of 10^6) missed all four small factors, although the method finds many others with these sizes. A two–step $P + 1$ algorithm found p_{16}; however, it missed p_{12} because the chosen value of $P^2 - 4$ was a quadratic residue modulo p_{12}. ECM found all four small prime factors easily.

One ECM factor found by CWI [25] is the 40–digit factor

$$1549314255\,0620385697\,1990677659\,9544873717$$

of $26^{126} + 1$. The job was run on the CRAY C90 at SARA, using 128 curves. The 102–nd curve had very smooth order:

$$2^9 \cdot 3 \cdot 1069 \cdot 1117 \cdot 11681 \cdot 14771 \cdot 55171 \cdot 142501 \cdot 154303 \cdot 4035751.$$

The largest factor found by ECM is the 43–digit factor

$$568\,8864305048\,6537027917\,5240510704\,4435136231$$

of the partition number $p(19997)$. It was found by Franz–Dieter Berger, University of Saarland (Germany), in 1993.

7. Algorithms for factoring arbitrary integers

The next several algorithms try to factor an odd integer N by finding two squares X^2 and Y^2 such that

$$X^2 \equiv Y^2 \pmod{N} \qquad \text{and} \qquad \gcd(XY,\, N) = 1. \tag{7.1}$$

Then they test $\gcd(X - Y,\, N)$, hoping for a non–trivial factor of N. Whereas the algorithms in §6 require time depending primarily on p to find the smallest prime factor p of N, the times for the upcoming algorithms depend primarily on the size of N itself.

If N has two distinct odd prime factors p_1 and p_2, and if X and Y are randomly selected subject to (7.1), then $\gcd(X - Y,\, N)$ will be non–trivial (i.e., neither 1 nor N) exactly 50% of the time. Indeed, choose Z such that $Z \equiv 1 \pmod{p_1}$ and $Z \equiv -1 \pmod{p_2}$; this Z exists by the Chinese remainder theorem. Then, given X, the solutions Y of $X^2 \equiv Y^2 \pmod{N}$ are $Y \equiv X$, $Y \equiv -X$, $Y \equiv XZ$, and $Y \equiv -XZ$. The corresponding values of $\gcd(X - Y,\, N)$ are N, 1, p_1 and p_2, respectively. Two of these four are non–trivial. If N has k distinct odd prime factors, then the probability of success with a single $(X,\, Y)$ pair is $1 - 2^{1-k}$.

These methods don't work if N is a prime power (i.e., if N doesn't have two distinct prime factors), but this condition is easily checked. If $N = p^k$ where p is prime and $k \geq 1$, then $N - 1 = p^k - 1$ is divisible by $p - 1$. By Fermat's little theorem, if $\gcd(a,\, N) = 1$, then

$$a^{N-1} = \left(a^{p-1}\right)^{(N-1)/(p-1)} \equiv 1^{(N-1)/(p-1)} \equiv 1 \pmod{p}.$$

Consequently $\gcd(a^{N-1} - 1,\, N)$ will be divisible by p. If instead we find an a such that $\gcd(a^N - a,\, N) = 1$, then n cannot be a prime power.

7.1 Finding squares through products

The best methods for constructing congruences of the form (7.1) start by accumulating several congruences

$$A_i \equiv B_i \pmod{N}, \tag{7.2}$$

where each A_i and each B_i is either a square or a square times a smooth number. These congruences are also called **relations**. For $N = 1098413$, a sample relation is

$$1100000 \equiv 1587 \pmod{N},$$

in which all prime divisors of $1100000 = 2^5 \cdot 5^5 \cdot 11$ and of $1587 = 3 \cdot 23^2$ are under 30.

The algorithms vary in how they find the relations (7.2). Once sufficiently many relations are found, each algorithm attempts to find a non–empty set S of indices such that both

$$\prod_{i \in S} A_i \qquad \text{and} \qquad \prod_{i \in S} B_i \tag{7.3}$$

are squares. The product of the corresponding congruences is a congruence of the form (7.1).

We illustrate with a small example, using $N = 77 = 7 \cdot 11$. The left side of Figure 7.1 lists some congruences modulo 77, in which the only prime factors of each side are 2, 3, 5. We deliberately suppress the congruences $81 \equiv 4$ and $256 \equiv 25$, where both sides are squares, since either of these factors 77 immediately.

We now multiply some of these congruences so as to generate squares on both sides. For example, $80 \cdot 320 \equiv 3 \cdot 243$ becomes $2^{10} \cdot 5^2 \equiv 3^6$. Rewrite this as $160^2 \equiv 27^2$. This congruence factors 77, since $\gcd(160 - 27,\, 77) = \gcd(133,\, 77) = 7$.

$45 \equiv$	-32
$50 \equiv$	-27
$72 \equiv$	-5
$75 \equiv$	-2
$80 \equiv$	3
$125 \equiv$	48
$320 \equiv$	243
$384 \equiv$	-1

	45	50	72	75	80	125	320	384
$p=2$	0	1	3	0	4	0	6	7
$p=3$	2	0	2	1	0	0	0	1
$p=5$	1	2	0	2	1	3	1	0
	-32	-27	-5	-2	3	48	243	-1
$p=-1$	1	1	1	1	0	0	0	1
$p=2$	5	0	0	1	0	4	0	0
$p=3$	0	3	0	0	1	1	5	0
$p=5$	0	0	1	0	0	0	0	0

Figure 7.1: Some congruences mod 77, involving powers of 2, 3, 5

We might instead have chosen to multiply $125 \equiv 48$ by $320 \equiv 243$. This gives $2^6 \cdot 5^4 \equiv 2^4 \cdot 3^6$, which is the same as $200^2 \equiv 108^2$. Since $\gcd(200-108,\ 77) = \gcd(92,\ 77) = 1$, this congruence does not yield a factorization.

The decision to multiply $80 \equiv 3$ by $320 \equiv 243$, or $125 \equiv 48$ by $320 \equiv 243$, is made by looking at the exponents of the primes in the resulting product. All exponents in $2^{10} \cdot 5^2 \equiv 3^6$ and in $2^6 \cdot 5^4 \equiv 2^4 \cdot 3^6$ are even. There are eight congruences in Figure 7.1; let the exponents e_1 to e_8 be one or zero, depending upon whether the corresponding congruence is included in or excluded from the product. The product congruence

$$\begin{aligned} &45^{e_1} \cdot 50^{e_2} \cdot 72^{e_3} \cdot 75^{e_4} \cdot 80^{e_5} \cdot 125^{e_6} \cdot 320^{e_7} \cdot 384^{e_8} \\ \equiv\ &(-32)^{e_1} \cdot (-27)^{e_2} \cdot (-5)^{e_3} \cdot (-2)^{e_4} \cdot 3^{e_5} \cdot 48^{e_6} \cdot 243^{e_7} \cdot (-1)^{e_8} \end{aligned} \tag{7.4}$$

factors into

$$\begin{aligned} &2^{e_2+3e_3+4e_5+6e_7+7e_8} \cdot 3^{2e_1+2e_3+e_4+e_8} \cdot 5^{e_1+2e_2+2e_4+e_5+3e_6+e_7} \\ \equiv\ &(-1)^{e_1+e_2+e_3+e_4+e_8} \cdot 2^{5e_1+e_4+4e_6} \cdot 3^{3e_2+e_5+e_6+5e_7} \cdot 5^{e_3}. \end{aligned}$$

Both sides will be squares precisely when all exponents are even. This is equivalent to requiring that all elements of the matrix–vector product

$$\begin{bmatrix} 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} e_1 \\ e_2 \\ e_3 \\ e_4 \\ e_5 \\ e_6 \\ e_7 \\ e_8 \end{bmatrix} \tag{7.5}$$

be even.

Equation (7.5) has the form $\mathbf{B}\mathbf{e} \equiv \mathbf{0} \pmod 2$, where $\mathbf{e}$ is the exponent vector and $\mathbf{B}$ is the 7×8 exponent matrix hidden in the right of Figure 7.1 but reduced modulo 2. The eight column vectors must be linearly dependent since all are in a space of dimension at most 7. This is equivalent to saying that there exists a nonzero $\mathbf{e} \in \mathrm{GF}(2)^8$ such that $\mathbf{B}\mathbf{e} = \mathbf{0}$.

In this example, the fifth, sixth, and seventh columns of $\mathbf{B}$ are all $[0,\ 0,\ 1,\ 0,\ 0,\ 1,\ 0]^{\mathrm{T}}$. Any two of these sum to zero modulo 2. This corresponds to multiplying two of the three congruences $80 \equiv 3$, $125 \equiv 48$, and $320 \equiv 243$, as we did earlier.

Another vector in the null space of $\mathbf{B}$ is $[1,\ 1,\ 0,\ 1,\ 0,\ 0,\ 1,\ 1]^{\mathrm{T}}$. The congruence

$$45 \cdot 50 \cdot 75 \cdot 320 \cdot 384 \equiv (-32) \cdot (-27) \cdot (-2) \cdot 243 \cdot (-1) \pmod{77}$$

becomes $144000^2 \equiv 648^2 \pmod{77}$, which again gives the factorization $77 = 7 \cdot 11$.

Traditionally, one solved the system $\mathbf{Be} = \mathbf{0}$ by a variation of Gaussian elimination. Recently some iterative methods [11, 12, 23, 38] have been found. The iterative methods are superior when the matrix is large, since they require less storage (matrices arising from integer factorization problems are very sparse). For these large, sparse, matrices, the iterative methods are also faster — if $\mathbf{B}$ is an $n \times n$ matrix, then Gaussian elimination uses $\mathcal{O}(n^3)$ bit operations but the iterative methods take $\mathcal{O}(n)$ applications of the matrix $\mathbf{B}$, which is time $\mathcal{O}(n^2)$ if the number of nonzero entries per column remains bounded as n grows.

7.2 Factor base

The set of primes appearing in the factorizations in Figure 7.1 is called the **factor base**. Often it is convenient to also include -1 in the factor base. If we allow primes below B to appear, then the size of the factor base is about $\pi(B)$ (see §3.5 for estimates of $\pi(B)$).

7.3 Free relations

In the last example, while factoring 77 with a factor base of $\{-1, 2, 3, 5\}$, we could have used the four trivial congruences

$$-1 \equiv -1, \qquad 2 \equiv 2, \qquad 3 \equiv 3, \qquad 5 \equiv 5 \tag{7.6}$$

in the product (7.4). For example, although $50 \cdot 75$ and $(-27) \cdot (-2)$ are not squares, the congruence

$$2 \cdot 3 \cdot 50 \cdot 75 \equiv 2 \cdot 3 \cdot (-27) \cdot (-2) \pmod{77}$$

yields $150^2 \equiv 18^2 \pmod{77}$ and a factorization. The congruences in (7.6) are called **free relations**, because the effort required to find the relations does not depend on the size of N.

In this example, which uses the factor base $\{-1, 2, 3, 5\}$ on both sides, we could dispense with the free relations and use the factorizations (including negative exponents) of the quotients $-45/32$, $-50/27$, $\cdots$ directly. Each rational quotient is congruent to 1, and the resulting matrix will be smaller since each prime appears only once (not once per side). The Number Field Sieve (§7.8) uses free relations which are more complicated than those shown here, and in which the two factor bases are different, so this simplification does not work there.

7.4 Continued fraction method

The continued fraction method (abbreviated CFRAC) is no longer in contention as a modern factoring method, but we include it because it is similar to some modern methods and easier to understand. It was used to factor the seventh Fermat number (39 digits) in 1970 [27, p. 184]:

$$2^{128} + 1 = 5964958\,9127497217 \cdot 57\,0468920068\,5129054721.$$

CFRAC looks for congruences $X^2 \equiv r \pmod{N}$ with small r (specifically $r = \mathcal{O}(\sqrt{N})$). For each congruence it finds, it attempts to factor r using the factor base. Where r is smooth, the congruence is saved so it can be multiplied by other such congruences to form squares on both sides.

If N is a perfect square, then it is easy to factor N. Otherwise $\sqrt{N}$ is irrational. There exist infinitely many rational approximations P/Q of $\sqrt{N}$ such that

$$\left|\frac{P}{Q} - \sqrt{N}\right| < \frac{1}{Q^2}.$$

If P/Q is any such approximation and we choose ϵ such that $P/Q = \sqrt{N} + \epsilon/Q^2$, then

$$P^2 - NQ^2 = \left(Q\sqrt{N} + \epsilon/Q\right)^2 - NQ^2 = 2\epsilon\sqrt{N} + \epsilon^2/Q^2.$$

Since $|\epsilon| < 1$, this shows that $|P^2 - NQ^2| < 2\sqrt{N} + 1/Q^2$. Hence all such values of $P^2 - NQ^2$ are $\mathcal{O}(\sqrt{N})$, as desired. We know a square root of $P^2 - NQ^2$ modulo N, namely P.

Convergent P/Q	$P^2 - N \cdot Q^2$
1049/1	$1988 = 2^2 \cdot 7 \cdot 71$
1048/1	$-109 = -109$
19913/19	$476 = 2^2 \cdot 7 \cdot 17$
80700/77	$-677 = -677$
181313/173	$1292 = 2^2 \cdot 17 \cdot 19$
262013/250	$-331 = -331$
1491378/1423	$1207 = 17 \cdot 71$
1753391/1673	$-796 = -2^2 \cdot 199$
3244769/3096	$1153 = 1153$
4998160/4769	$-493 = -17 \cdot 29$
18239249/17403	$1084 = 2^2 \cdot 271$
23237409/22172	$-911 = -911$
41476658/39575	$839 = 839$
64714067/61747	$-1228 = -2^2 \cdot 307$
106190725/101322	$133 = 7 \cdot 19$

Table 7.1: Approximations to $\sqrt{1098413}$

As an example, with $N = 1098413$, the first 15 continued fraction approximations to $\sqrt{N}$ appear in Table 7.1. Three values of $P^2 - NQ^2$ are

$$\begin{aligned} 19913^2 - N \cdot 19^2 &= 476 = 2^2 \cdot 7 \cdot 17, \\ 181313^2 - N \cdot 173^2 &= 1292 = 2^2 \cdot 17 \cdot 19, \\ 106190725^2 - N \cdot 101322^2 &= 133 = 7 \cdot 19. \end{aligned} \tag{7.7}$$

The product of the three right sides in (7.7) is a square, namely $2^4 \cdot 7^2 \cdot 17^2 \cdot 19^2$. Multiply the three left sides and suppress the multiples of N to reveal

$$(19913 \cdot 181313 \cdot 106190725)^2 \equiv \left(2^2 \cdot 7 \cdot 17 \cdot 19\right)^2 \pmod{N}.$$

Reduce each parenthesized argument modulo N to get $9044^2 \equiv 9044^2 \pmod{N}$. Unfortunately, this trivial congruence does not yield a factorization. We could try more, but will instead illustrate other algorithms.

Table 7.1 gives values of P and Q to full precision. The recurrences for P and Q allow one to work with $P \bmod N$ and $Q \bmod N$ instead of P and Q themselves, and can be evaluated quickly. For example, the numerator and denominator of the last entry in Table 7.1 are the sums of those parts of the two previous entries.

Some small primes (e.g., 3, 5, 11) are missing in Table 7.1. This is because they are not quadratic residues modulo N. If $p|(P^2 - NQ^2)$ where $\gcd(P, Q) = 1$, then N must be a quadratic residue modulo p unless $p \mid N$. Unless N is a perfect square, only half of the primes (asymptotically) have N as a quadratic residue. If B is the upper limit on the factor base, then the factor base size is about $\pi(B)/2$ rather than $\pi(B)$.

7.5 Sieving

Much of the time in CFRAC is spent factoring the residues $P^2 - NQ^2$, to test whether they are smooth. This work is done primarily by trial division, although one may employ the other methods in this survey too.

Quadratic Sieve (see §7.6) eliminates this burden. If $f \in \mathbb{Z}[X]$ is a univariate polynomial with integer coefficients, and p is a prime, then the values of x for which $p \mid f(x)$ lie in a few arithmetic progressions. By (3.2), if k is an integer, $f(x + kp) \equiv f(x) \pmod{p}$. Therefore $f(x + kp)$ will be divisible by p if and only if $f(x)$ is divisible by p.

Suppose we want to evaluate a polynomial f at several consecutive values of x and check each value for smoothness. Start by building a table of values of $f(x)$. For each prime p in our factor base, find the roots of f modulo p, by factoring $f(X)$ over $\mathrm{GF}(p)$ [16, §4.6.2]. Then, for each x such that $f(x) \equiv 0 \pmod{p}$, replace our tabulated value of $f(x)$ by $f(x)/p$. After processing all primes in our factor base, if any table entry is ± 1, then the corresponding $f(x)$ was smooth.

This procedure can be improved considerably. One improvement tabulates $\log|f(x)|$ rather than $f(x)$, and subtracts $\log p$ rather than dividing by p. The logarithms can be approximate, perhaps to base 2. At the end, look for small values in the table, not just for a value of $\log 1 = 0$. This procedure will also find some values of x for which $f(x)$ is smooth but not squarefree (i.e., for which a prime power divides $f(x)$).

7.6 Quadratic sieve

Using the ideas in the last section, Quadratic Sieve [31] looks at the values of a quadratic polynomial at successive points. We illustrate it by a detailed example. Define $f(X) = X^2 - N$, where $N = 1098413$. After sieving $f(x)$ for values of x near $\lfloor\sqrt{N}\rfloor = 1048$, we accumulate data similar to that in Figure 7.2.

$$
\begin{aligned}
f(\;925) &= -2^2 \cdot 7 \cdot 13 \cdot 23 \cdot 29 \\
f(1047) &= -2^2 \cdot 19 \cdot 29 \\
f(1051) &= 2^2 \cdot 7 \cdot 13 \cdot 17 \\
f(1063) &= 2^2 \cdot 7^3 \cdot 23 \\
f(1077) &= 2^2 \cdot 7 \cdot 13^3 \\
f(1119) &= 2^2 \cdot 7 \cdot 17^2 \cdot 19 \\
f(1142) &= 7^2 \cdot 13 \cdot 17 \cdot 19
\end{aligned}
$$

	925	1047	1051	1063	1077	1119	1142
$p = -1$	1	1	0	0	0	0	0
$p = 2$	0	0	0	0	0	0	0
$p = 7$	1	0	1	1	1	1	0
$p = 13$	1	0	1	0	1	0	1
$p = 17$	0	0	1	0	0	0	1
$p = 19$	0	1	0	0	0	1	1
$p = 23$	1	0	0	1	0	0	0
$p = 29$	1	1	0	0	0	0	0

Figure 7.2: Smooth values of $f(X) = X^2 - 1098413$ and associated binary matrix

The third, sixth, and seventh columns in the right table of Figure 7.2 sum to zero modulo 2. Hence the product

$$f(1051)f(1119)f(1142) = (2^2 \cdot 7 \cdot 13 \cdot 17)\,(2^2 \cdot 7 \cdot 17^2 \cdot 19)\,(7^2 \cdot 13 \cdot 17 \cdot 19)$$
$$= 2^4 \cdot 7^4 \cdot 13^2 \cdot 17^4 \cdot 19^2.$$

gives a square on the right. Take square roots (and recall the definition of f) to derive:

$$(1051 \cdot 1119 \cdot 1142)^2 \equiv (2^2 \cdot 7^2 \cdot 13 \cdot 17^2 \cdot 19)^2 \pmod{N}.$$

A calculation gives $1051 \cdot 1119 \cdot 1142 \equiv 810112 \pmod{N}$ and $2^2 \cdot 7^2 \cdot 13 \cdot 17^2 \cdot 19 \equiv 810112 \pmod{N}$. Unfortunately the congruence $810112^2 \equiv 810112^2 \pmod{N}$ does not help.

Multiple Polynomials. If we sieve the $2M$ values of $f(x)$ for $\left|x - \sqrt{N}\right| \le M$, then the largest residue is about $2M\sqrt{N}$ (assuming $M \ll \sqrt{N}$). Montgomery [36] found a way to stunt this growth as M grows. His variation is called the **Multiple Polynomial Quadratic Sieve**, or **MPQS**.

Let $k = 1$ if $N \equiv 1 \pmod 4$ and $k = 4$ if $N \equiv 3 \pmod 4$. Find a quadratic polynomial $g(X) = a^2X^2 + bX + c$ such that $b^2 - 4a^2c = kN$. For example, when $N = 1098413$ and $k = 1$, we might pick

$$g(X) = 841X^2 + 293X - 301. \tag{7.8}$$

We discuss how to choose g below. Once g is selected, we sieve to find values of x for which $g(x)$ is smooth. In this case both

$$g(-1) = 247 = 13 \cdot 19 \qquad \text{and} \qquad g(1) = 833 = 7^2 \cdot 17$$

are smooth. Because

$$g(X) = \left(aX + \frac{b}{2a}\right)^2 - \frac{b^2 - 4a^2c}{4a^2} \equiv \left(aX + \frac{b}{2a}\right)^2 \pmod{N},$$

the square roots of 247 and 833 modulo 1098413 are $-29 + 293/58 = -1389/58$ and $29 + 293/58 = 1975/58$, respectively. These can be merged with other data in Figure 7.2 to produce squares on both sides. One such product

$$\left(1051 \cdot 1077 \cdot \frac{1975}{58}\right)^2 \equiv f(1051)f(1077)g(1) = 2^4 \cdot 7^4 \cdot 13^4 \cdot 17^2 \pmod{N}$$

yields $838199^2 \equiv 563108^2 \pmod{N}$, which factors N.

The polynomial $g(X)$ in (7.8) can be found by first selecting an odd prime value for a (here 29). We require that kN be a quadratic residue modulo a. Solve $b_0^2 \equiv kN \pmod{a}$ for b_0. Then solve $(b_0 + \ell a)^2 \equiv kN \pmod{a^2}$ for ℓ. Set $b = b_0 + \ell a$ or $b = b_0 + \ell a - a^2$, whichever has the same parity as kN. Define $c = (b^2 - kN)/4a^2$. By construction, c is an integer and $b^2 - 4a^2c = kN$.

When sieving over $|x| \le M_0$, the values of a, b, c should be picked so that the values $|g(-M_0)|$, $|g(0)|$, and $|g(M_0)|$ are approximately equal. That is, the parabola should cross the x–axis twice in the interval $[-M_0, M_0]$ and the three extrema should have comparable magnitudes. The solution (subject to $b^2 - 4a^2c = kN$) is

$$a^2 \approx \frac{\sqrt{kN/2}}{M_0}, \qquad b \approx 0, \qquad c \approx -M_0\sqrt{kN/8}\,.$$

The largest polynomial value is about $|c|$, or $M_0\sqrt{kN/8}$, which is at most $M_0\sqrt{N/2}$. To sieve $2M$ values of x, one can use M/M_0 different polynomials, sieving $2M_0$ values per polynomial. The largest residual is $\mathcal{O}(M_0\sqrt{N})$ rather than $\mathcal{O}(M\sqrt{N})$. Details are in [36].

Since values from different polynomials can be combined, the sieving portion of the MPQS algorithm is easily parallelized. Each processor sieves different polynomials, and all smooth residues go to a central site. This was used for the RSA–129 factorization mentioned in §8.1.

As in CFRAC, the only primes in the factor base are those for which kN is a quadratic residue.

7.7 Large Prime Variations

The sieving procedure in §7.5 looks for values of x such that $f(x)$ is smooth with respect to the factor base. The algorithm is easily modified to also find values of x for which $f(x)$ is a smooth number times a prime not much larger than the factor base bound, by adjusting the threshold used when inspecting logarithms after sieving. The extra prime in the factorization of $f(x)$ is called a **large prime**. If one finds two values of x for which $f(x)$ has the same large prime, then the corresponding congruences can be multiplied together and treated as a pair for the rest of the algorithm.

This procedure, called the **large prime variation**, is compatible with the use of multiple polynomials described in the last section. For example, both $f(1040) = -17 \cdot 23 \cdot 43$ and $g(0) = -7 \cdot 43$ have 43 as the only prime exceeding 29. After doing the linear algebra phase, we decide to combine these with three entries in Figure 7.2 to get the product

$$\begin{aligned} & g(0)f(1040)f(1051)f(1063)f(1077) \\ &= (-7 \cdot 43)\,(-17 \cdot 23 \cdot 43)\,(2^2 \cdot 7 \cdot 13 \cdot 17)\,(2^2 \cdot 7^3 \cdot 23)\,(2^2 \cdot 7 \cdot 13^3) \\ &= 2^6 \cdot 7^6 \cdot 13^4 \cdot 17^2 \cdot 23^2 \cdot 43^2. \end{aligned}$$

This gives the congruence

$$\left(\frac{293}{58} \cdot 1040 \cdot 1051 \cdot 1063 \cdot 1077\right)^2 \equiv \left(2^3 \cdot 7^3 \cdot 13^2 \cdot 17 \cdot 23 \cdot 43\right)^2,$$

which simplifies to $970009^2 \equiv 257894^2$ and factors N.

Another variation of MPQS uses two large primes instead of one; this version is known as **PPMPQS**. See [19].

7.8 Number Field Sieve

The Number Field Sieve (NFS) [17, 18] uses ideas from algebraic number theory. It made newspaper headlines in 1990 when it was used to factor the 148–digit cofactor $(2^{512} + 1)/2424833$ of the ninth Fermat number[18].

Suppose N is a composite integer to be factored. NFS has four main phases:

> **Polynomial selection.** Select two irreducible univariate polynomials $f(X)$ and $g(X)$ with "small" integer coefficients for which there exists an integer m such that
>
> $$f(m) \equiv g(m) \equiv 0 \pmod{N}.$$
>
> The polynomials f and g should not have a common factor over $\mathbb{Q}$. Often one polynomial is $X - m$, and the other has the coefficients of the base–m expansion of N, for suitable m.
>
> There is no known "good" way to pick these polynomials, unless our original number has a special algebraic form such as the $(12^{151} - 1)/11$ shown in §8.2. For the ninth

Fermat number, the polynomials were chosen to be $X - 2^{103}$ and $X^5 - 8$, with common root $m = 2^{103}$.

Let α denote a (complex) root of f and β denote a root of g.

Sieving. This phase finds pairs (a, b) such that $\gcd(a, b) = 1$ and such that both

$$b^{\deg(f)} f(a/b) \qquad \text{and} \qquad b^{\deg(g)} g(a/b) \tag{7.9}$$

are smooth with respect to a chosen factor base.

The sieving phase can fix b and search for values of a such that both polynomial functions in (7.9) are smooth, using the ideas in §7.5. Although we require two values be smooth (rather than one value, as in MPQS), the values in (7.9) are sufficiently smaller that we gain overall.

Linear algebra. The expressions in (7.9) are the norms of the algebraic numbers $a - b\alpha$ and $a - b\beta$, multiplied by the leading coefficients of f and of g, respectively. The principal ideals $(a - b\alpha)$ and $(a - b\beta)$ factor into products of prime ideals in the number fields $\mathbb{Q}(\alpha)$ and $\mathbb{Q}(\beta)$, respectively. All prime ideals appearing in these factorizations have small norm (since the norms are assumed to be smooth), so only a few different prime ideals can appear in these factorizations. Use linear algebra to find a set S of indices such that the two products

$$\prod_{i \in S} ((a_i - b_i\alpha)) \qquad \text{and} \qquad \prod_{i \in S} ((a_i - b_i\beta)) \tag{7.10}$$

are both squares of products of prime ideals.

Square root. Using the set S in (7.10), try to find algebraic numbers $\alpha' \in \mathbb{Q}(\alpha)$ and $\beta' \in \mathbb{Q}(\beta)$ such that

$$(\alpha')^2 = \prod_{i \in S} (a_i - b_i\alpha) \qquad \text{and} \qquad (\beta')^2 = \prod_{i \in S} (a_i - b_i\beta).$$

Couveignes's algorithm [13] works if the polynomials f and g have odd degrees; Montgomery's square root algorithm [24] allows arbitrary degree.

Let $\phi_\alpha : \mathbb{Q}(\alpha) \to \mathbb{Z}/N\mathbb{Z}$ and $\phi_\beta : \mathbb{Q}(\beta) \to \mathbb{Z}/N\mathbb{Z}$ be homomorphisms induced by setting $\phi_\alpha(\alpha) = \phi_\beta(\beta) = m$, where m is the common root of f and g. The congruence

$$\phi_\alpha(\alpha')^2 = \phi_\alpha\left((\alpha')^2\right) = \phi_\alpha\left(\prod_{i \in S} (a_i - b_i\alpha)\right) \equiv \prod_{i \in S} (a_i - b_i m) \equiv \phi_\beta(\beta')^2 \pmod{N}$$

has the form (7.1); the two sides will be coprime to N if none of the factorizations in (7.9) share a factor with N.

Example of NFS. The first step in NFS is polynomial selection. If we somehow observe that

$$N = 1098413 = 1093500 + 4913 = 12 \cdot 45^3 + 17^3,$$

then we can choose

$$f(X) = X^3 + 12 \qquad \text{and} \qquad g(X) = 45X - 17.$$

Both polynomials vanish modulo N when $X \equiv 17/45 \equiv 634639 \pmod{1098413}$. After sieving and linear algebra, we construct the product

$$h(X) = -(X-6)(2X+3)(3X-7)(3X+1)(5X-2)(8X-3)(10X+9). \tag{7.11}$$

We claim that (7.11) gives squares on both sides. More precisely, with $\alpha = \sqrt[3]{-12}$ and $\phi_\alpha(\alpha) = 17/45$,

$$h\left(\frac{17}{45}\right) = \frac{2^8 \cdot 11^2 \cdot 13^2 \cdot 23^2}{3^{12} \cdot 5^4} = \left(\frac{52624}{18225}\right)^2$$

and

$$h(\alpha) = 7400772 + 1138236\alpha - 105495\alpha^2 = (\alpha')^2, \qquad \text{where}$$

$$\begin{aligned} \alpha' &= 2694 + 213\alpha - 28\alpha^2, \\ \phi_\alpha(\alpha') &= \frac{5610203}{2025}. \end{aligned}$$

The factor 1951 divides the numerator of $\dfrac{5610203}{2025} - \dfrac{52624}{18225} = \dfrac{50439203}{18225}$.

When selecting (7.11), we included a factor of -1 on both sides. The congruence $-1 \equiv -1$ is a free relation, much as in §7.3. There is also one free relation for each prime p such that the polynomials $f(X)$ and $g(X)$ both split completely modulo p, but no such relations were used in (7.11).

8. Improvements in technology

8.1 RSA–129 factorization

In a 1977 MIT technical memo, which Martin Gardner summarizes in his *Mathematical Games* column[14], Rivest et al challenge the public to factor a 129–digit which they claim is the product of 64–digit and 65–digit factors. Rivest estimates that the required running time, using the best algorithms and machines available in 1977, would be 40 quadrillion years. It took much less time than predicted. After a 8–month worldwide effort [20] organized by Derek Atkins, Michael Graff, Arjen Lenstra, and Paul Leyland, the factorization was completed by PPMPQS in April, 1994. This effort took an estimated 5000 MIPS years. It found

$$\begin{aligned} \text{RSA-129} = {} & 114381625\,7578888676\,6923577997\,6146612010\,2182967212 \\ & 4236256256\,1842935706\,9352457338\,9783059712 \\ & 3563958705\,0589890751\,4759929002\,6879543541, \\ = {} & p_{64} \cdot p_{65}, \qquad \text{where} \\ p_{64} = {} & 3490\,5295108476\,5094914784 \\ & 9619903898\,1334177646\,3849338784\,3990820577, \\ p_{65} = {} & 32769\,1329932667\,0954996198 \\ & 8190834461\,4131776429\,6799294253\,9798288533. \end{aligned}$$

8.2 Factorizations found at CWI

In June, 1994, researchers at CWI and in Oregon, USA achieved some record factorizations using the number field sieve.

The first was the 162–digit Cunningham number $N_{162} = (12^{151} - 1)/11$. No factors were known. At Oregon State University (OSU) in USA, Peter Montgomery et al had sieved this number using NFS with the two polynomials

$$12X^5 - 1 \qquad \text{and} \qquad X - 12^{30}.$$

They used about 30 workstations at OSU over an 8–week period during spring and summer, 1993. The researchers gathered 8.98 million relations, but were unable to process the data and find the factorization. During 1993–1994, while Montgomery was at CWI, the Computational Number Theory group at CWI completed the linear algebra and square root phases of the work. They found the factorization $N_{162} = p_{44} \cdot p_{119}$, where

$$\begin{aligned}
N_{162} = {}& 82\,2196205286\,5970195266\,0120743076\,1004273909 \\
& 2435707339\,6551677033\,9373353207\,4305023580 \\
& 2427303275\,6332005408\,0668946066\,9679221954 \\
& 5093967127\,3308456244\,6289606063\,0268212317, \\
p_{44} = {}& 1653\,7237851564\,6889242614\,0704164885\,3990657743, \\
p_{119} = {}& 497178678\,0032337881\,8763399005\,9600164874 \\
& 7659834953\,9211569747\,0057591532\,2824191116 \\
& 7043200927\,0168842857\,3103024883\,1349126419.
\end{aligned}$$

The special algebraic form of N_{162} simplified the polynomial selection phase. This beat the 158–digit record, which A.K. Lenstra and Dan Bernstein had previously achieved using NFS.

The OSU team also sieved the following 105–digit cofactor of $3^{367} - 1$:

$$\begin{aligned}
N_{105} = {}& 75870\,1086707710\,3419834518 \\
& 9863846063\,0208179089\,1150247368\,3674638356 \\
& 7258455011\,6888623834\,4212966512\,3030350997.
\end{aligned}$$

Using the data gathered at OSU, the CWI group found $N_{105} = p_{52} \cdot p_{54}$, where

$$\begin{aligned}
p_{52} = {}& 15\,1149525784 \\
& 0070716998\,8656940229\,3793503992\,8231350493, \\
p_{54} = {}& 5019\,5399738924 \\
& 4528404247\,9062790906\,5410546896\,2124251929.
\end{aligned}$$

This time the polynomial selection phase was more complicated. The researchers used two quadratic polynomials:

$$\begin{aligned}
f(X) = {}& 34\,2910527737\,X^2 + 868170\,6933351946\,5483641612\,X \\
& +540759062\,6047829713\,5713953618\,6424874771, \\
g(X) = {}& 124\,2060255079\,X^2 - 9130492\,7318176881\,6962553218\,X \\
& +12\,9128767300\,0652336311\,6822953626\,7982420800.
\end{aligned}$$

The resultant of these polynomials is $9N_{105}$, so they share a common root modulo N.

The N_{105} was the first large number completed by NFS which did not have a special form. The record was broken a month later when three researchers completed a 116–digit cofactor of the partition number $p(11887)$, using a fifth–degree polynomial and a linear polynomial.

The polynomial selection, sieving, and linear algebra phases were done by Arjen Lenstra and Bruce Dodson in the USA; the square root phase was done by Peter Montgomery at CWI.

For N_{162}, the factor bases had all primes below 2 million (on small workstations) or below 3.5 million (on SPARC 10's). The program allowed two large primes up to 100 million on each side. The (sparse) matrix was 828077×833017 with an average of 32.3 nonzero entries per column.

For N_{105}, the factor bases contained all primes below 1.6 million and large primes went to 30 million. The sieving was performed in such a way that every relation contained at least one large prime between 20 million and 30 million, and could contain five large primes. The matrix was 1284719×1294861 with an average of 30.1 nonzero entries per column

These matrices are larger than any previous matrices arising from integer factorization problems. The N_{105} matrix would require 200 gigabytes of memory to store in dense form, which is more than most sites have even on secondary storage. This prevented the Oregon researchers from finishing the work. The CWI researchers used a novel Block Lanczos algorithm [23] for the linear algebra phase, and completed the larger problem in 7.5 hours on a Cray C90 at the Academic Computing Center Amsterdam (SARA).

9. Acknowledgements

This work was funded by Centrum voor Wiskunde en Informatica (Amsterdam) and by the Stieltjes Institute for Mathematics (Leiden). The manuscript was revised while the author visited Bellcore (USA). Thanks to Richard Brent, Mary Flahive, Marije Huizing, Arjen Lenstra, and Herman teRiele for reviewing early drafts of this manuscript.

References

1. Th. Beth, M. Frisch, and G.J. Simmons, editors. *Public–Key Cryptography: State of the Art and Future Directions*, volume 578 of *Lecture Notes in Computer Science*. Springer–Verlag, Berlin Heidelberg, 1992. Report on workshop at Oberwolfach, Germany, July, 1991.

2. Richard P. Brent. An improved Monte Carlo factorization algorithm. *BIT*, 20(2):176–184, 1980.

3. Richard P. Brent. Some integer factorization algorithms using elliptic curves. Research Report CMA–R32–85, Centre for Mathematical Analysis, The Australian National University, GPO Box 4, Canberra, ACT 2601, Australia, September 1985.

4. Richard P. Brent and John M. Pollard. Factorization of the eighth Fermat number. *Mathematics of Computation*, 36(154):627–630, April 1981.

5. R.P. Brent and H.J.J. te Riele. Factorizations of $a^n \pm 1$, $13 \leq a \leq 100$. Report NM–R9212, CWI Amsterdam, June 1992.
   Available by anonymous ftp from `nimbus.anu.edu.au:pub/Brent/rpb134t.txt.Z, rpb134.dvi.Z` .
   Update 1 to this report has appeared as CWI Report NM–R9419, September 1994, with P.L. Montgomery as another author.

6. David M. Bressoud. *Factorization and Primality Testing*. Springer–Verlag, New York, NY, 1989. Undergraduate Texts in Mathematics.

7. John Brillhart, D.H. Lehmer, J.L. Selfridge, Bryant Tuckerman, and S.S. Wagstaff, Jr. *Factorization of* $b^n \pm 1$, $b = 2, 3, 5, 6, 7, 10, 11, 12$ *up to high powers*, volume 22 of *Contemporary Mathematics*. American Mathematical Society, Providence, RI, 2nd edition,

1988.

8. John Brillhart, Peter L. Montgomery, and Robert D. Silverman. Tables of Fibonacci and Lucas factorizations. *Mathematics of Computation*, 50(181):251–260 & S1–S15, January 1988.
9. John Brillhart and J.L. Selfridge. Some factorizations of $2^n \pm 1$ and related results. *Mathematics of Computation*, 21(97):87–96, January 1967. Corrigendum, *ibid.*, p. 751.
10. Henri Cohen. *A Course in Computational Algebraic Number Theory*, volume 138 of *Graduate Texts in Mathematics*. Springer–Verlag, Berlin, 1993.
11. Don Coppersmith. Solving linear equations over GF(2): Block Lanczos algorithm. *Linear Algebra and its Applications*, 192:33–60, October 1993.
12. Don Coppersmith. Solving homogeneous linear equations over GF(2) via block Wiedemann algorithm. *Mathematics of Computation*, 62(205):333–350, January 1994.
13. Jean-Marc Couveignes. Computing a square root for the number field sieve. In A.K. Lenstra and H.W. Lenstra, Jr., editors, *The Development of the Number Field Sieve*, volume 1554 of *Lecture Notes in Mathematics*, pages 95–102. Springer–Verlag, Berlin, 1993.
14. Martin Gardner. Mathematical games. A new kind of cipher that would take millions of years to break. *Scientific American*, 237(2):120–124, August 1997.
15. Richard K. Guy. How to factor a number. In B.L. Hartnell and H.C. Williams, editors, *Congressus Numerantium XVI: Proceedings of the Fifth Manitoba Conference on Numerical Mathematics*, pages 49–89, Winnipeg, 1976. Utilitas Mathematica Publishing Incorporated.
16. Donald E. Knuth. *Seminumerical Algorithms*, volume 2 of *The Art of Computer Programming*. Addison–Wesley, Reading, MA, 2nd edition, 1981.
17. A.K. Lenstra and H.W. Lenstra, Jr., editors. *The Development of the Number Field Sieve*, volume 1554 of *Lecture Notes in Mathematics*. Springer–Verlag, Berlin, 1993.
18. A.K. Lenstra, H.W. Lenstra, Jr., M.S. Manasse, and J.M. Pollard. The factorization of the ninth Fermat number. *Mathematics of Computation*, 61(203):319–349, July 1993.
19. A.K. Lenstra and M.S. Manasse. Factoring with two large primes. *Mathematics of Computation*, to appear.
20. Arjen K. Lenstra and Mark S. Manasse. Factoring by electronic mail. In J.-J. Quisquater and J. Vandewalle, editors, *Advances in Cryptology — EUROCRYPT '89*, volume 434 of *Lecture Notes in Computer Science*, pages 355–371, Berlin, 1990. Springer–Verlag.
21. H.W. Lenstra, Jr. Factoring integers with elliptic curves. *Annals of Mathematics*, 126(3):649–673, November 1987. Second series.
22. Peter L. Montgomery. Speeding the Pollard and elliptic curve methods of factorization. *Mathematics of Computation*, 48(177):243–264, January 1987.
23. Peter L. Montgomery. A block Lanczos algorithm for finding dependencies over GF(2). Technical report, CWI Amsterdam, 1994. To appear.
24. Peter L. Montgomery. Square roots of products of algebraic numbers. In Walter Gautschi, editor, *Mathematics of Computation 1943–1993: a Half-Century of Computational Math-*

ematics. Proceedings of Symposia in Applied Mathematics, American Mathematical Society, 1994. To appear.

25. Peter L. Montgomery. Vectorization of the elliptic curve method. Technical report, CWI Amsterdam, 1994. To appear.
26. Peter L. Montgomery and Robert D. Silverman. An FFT extension to the $P-1$ factoring algorithm. *Mathematics of Computation*, 54(190):839–854, April 1990.
27. Michael A. Morrison and John Brillhart. A method of factoring and the factorization of F_7. *Mathematics of Computation*, 29(129):183–205, January 1975.
28. J.M. Pollard. Theorems on factorization and primality testing. *Proc. Camb. Phil. Soc.*, 76(2):521–528, September 1974.
29. J.M. Pollard. A Monte Carlo method for factorization. *BIT*, 15(3):331–334, 1975.
30. Carl Pomerance. Analysis and comparison of some integer factoring algorithms. In H.W. Lenstra, Jr. and R. Tijdeman, editors, *Computational Methods in Number Theory, Part I*, pages 89–130. Mathematisch Centrum, Amsterdam, 1982. Math. Centrum Tract 154.
31. Carl Pomerance. The quadratic sieve factoring algorithm. In T. Beth, N. Cot, and I. Ingemarsson, editors, *Advances in Cryptology, Proceedings of EUROCRYPT 84*, volume 209 of *Lecture Notes in Computer Science*, pages 169–182, New York, 1985. Springer–Verlag.
32. Hans Riesel. *Prime Numbers and Computer Methods for Factorization*, volume 57 of *Progress in Mathematics*. Birkhäuser, Boston, 1985.
33. R.L. Rivest, A. Shamir, and L. Adleman. A method for obtaining digital signatures and public–key cryptosystems. *CACM*, 21(2):120–126, February 1978.
34. RSA Challenge Administrator. Information about RSA Factoring Challenge, March 1991. Send electronic mail to challenge–info@rsa.com.
35. Joseph H. Silverman. *The Arithmetic of Elliptic Curves*, volume 106 of *Graduate Texts in Mathematics*. Springer–Verlag, New York, 1986.
36. Robert D. Silverman. The multiple polynomial quadratic sieve. *Mathematics of Computation*, 48(177):329–339, January 1987.
37. Robert D. Silverman. Massively distributed computing and factoring large integers. *CACM*, 34(11):95–103, November 1991.
38. Douglas H. Wiedemann. Solving sparse linear equations over finite fields. *IEEE Trans. Inform. Theory*, 32(1):54–62, January 1986.
39. H.C. Williams. A $p+1$ method of factoring. *Mathematics of Computation*, 39(159):225–234, July 1982.
40. H.C. Williams and J.O. Shallit. Factoring integers before computers. In Walter Gautschi, editor, *Mathematics of Computation 1943–1993: a Half-Century of Computational Mathematics*. Proceedings of Symposia in Applied Mathematics, American Mathematical Society, 1994. To appear.