Computational Number Theory

H.J.J. te Riele

1. Introduction

Natural numbers—the numbers by which we count—have always struck the imagination. Problems involving natural numbers often are simple to state, but their solution may require a lot of mathematical creativity and ingenuity. A classical example is the problem of perfect numbers which was studied already by the ancient Greeks: nowadays, it has become a standard problem for testing the accuracy and reliability of new computers and software. (See also figure 1.) Daily life, even in antiquity, is unthinkable without counting, so the scientific discipline now known as *Number Theory* finds its roots in practical problems of every day.

In number theory, one studies the properties of natural and rational numbers and the solution of equations by such numbers. Some typical questions are: what are the divisors of a given number and how many are there? How many prime numbers (i.e., numbers > 1 only divisible by 1 and themselves) are there below a given bound x? Is there an $n > 2$ for which the equation $x^n + y^n = z^n$ has a solution in rational numbers?

Many problems can be solved in a step-by-step way, i.e., with the help of an *algorithm.* Loosely speaking, an algorithm is a set of arithmetic rules which yields, when applied to a prescribed input, a definite output in a finite number of computational steps. The invention of mechanical and electronic computers meant a huge step forward for the study of algorithms. These machines, if programmed correctly, do not make mistakes nor lose concen-

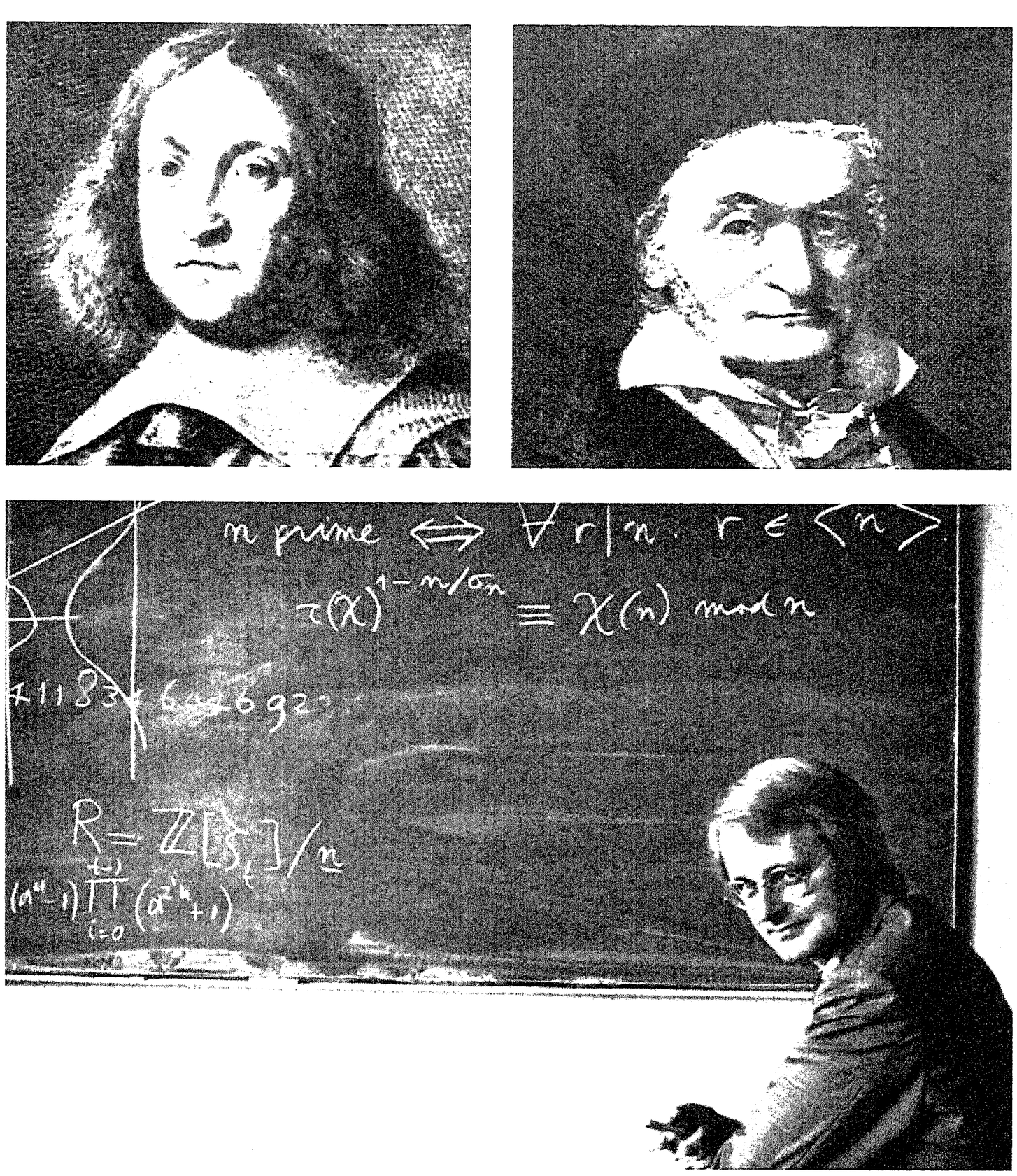

Figure 1. Factorization of numbers has fascinated mathematicians from ancient Greece through to the present day. Three prominent representatives are: Pierre de Fermat (1601-1665, above left), C.F. Gauss (1777-1855, above right), and Hendrik W. Lenstra, Jr. (b. 1949, below).

tration, so they are perfectly suited to serve as modern slaves for the tedious and time-consuming work of the computational number-theorist. In addition, many known ingenious algorithms for answering number-theoretic

questions would not have been invented *without* computers. For example, modern algorithms for finding the prime factors of *large* numbers, are very inefficient for *small* numbers, and it is even practically impossible to apply them without the help of fast electronic computers. The advent of vector computers in the eighties, and parallel computers in the nineties has stimulated the study of algorithms for such architectures. Several number-theoretic problems, like those where one wants to find numbers with a special property, are well-suited for treatment with the help of such vector and/or parallel algorithms.

These developments have given rise to the birth of 'Computational (or Algorithmic) Number Theory'. Here, the computer is a tool for experimentation and for testing hypotheses, and it is a stimulus for the development of ever more efficient algorithms, which can lead to new insight and new mathematical results.

2. GLOBAL DEVELOPMENTS IN COMPUTATIONAL NUMBER THEORY

Before the advent of fast electronic computers, *tables* were an important aid to number-theorists. Nowadays, it is much more efficient to save a *computer program* or *implemented algorithm* and to quickly generate the tables or individual table entries each time they are needed. Collections of such (sub)programs are available now in several *computer algebra packages* like PARI, MAPLE and MATHEMATICA. They enable the researcher to perform arithmetic calculations (in arbitrary precision) on mathematical objects such as numbers, vectors, matrices, algebraic numbers and finite fields, and to perform *symbolic* computation like integration, differentiation and formal series expansion.

The design, implementation, and analysis of efficient algorithms for solving number-theoretic problems has been the main activity of researchers in computational number theory. As an illustration, we will briefly describe here three major algorithmic developments, namely, in factoring, primality testing and lattice basis reduction. A survey of modern factoring methods can be found in [2]. An excellent historical survey of the computational history of factoring and primality testing from 1750 to about 1950, i.e., *before the era of electronic computers* is presented in [5]. Old and modern primality tests are treated in [1]; this book also treats algorithms for lattice basis reduction. An excellent textbook on algorithmic *algebraic* number theory is [3].

2.1. Factoring

An important stimulus for the study of *factoring* algorithms was the discovery by R.L. Rivest, A. Shamir and L. Adleman, in 1978, of a public-key encryption scheme, now known as RSA. It is based on the (presumed) difficulty of decomposing a given large number into prime factors. For the cur-

rently best known factoring algorithms practical experience suggests that factoring indeed is a hard problem, although nothing has been proved so far.

The best known factoring algorithms try to find integers x and y with the property that

$$x^2 \equiv y^2 \bmod n, \tag{2.1}$$

where n is the number to be factored (and known to be composite). For such x and y, the number $d = \gcd(x - y, n)$ is easily computed by a well-known algorithm of Euclid, and d is a proper divisor of n in at least half the number of cases for which (2.1) holds. So if we have no success, we try to find another pair x, y. To find a congruence of the form (2.1), one tries to collect *many* congruences of the form $x_i^2 \equiv a_i \bmod n$, where the a_i only have prime factors in a given set $\mathcal{F}$, which is called the *factor base*. If we have succeeded to find more such congruences (also called *relations*) than there are different primes in the factor base, we can combine them with the help of linear algebra techniques (Gaussian elimination or iterative methods), to find a congruence of the form (2.1). There are several methods to find the above relations. One is based on the computation of the continued fraction of $\sqrt{n}$ and another is based on efficient sieving techniques for finding values of quadratic polynomials which only consist of prime factors in the set $\mathcal{F}$.

Two important algorithmic discoveries have effectuated a jump in the size of the numbers which can be factored within a reasonable time on a modern computer: the quadratic sieve method (QS) published in its modern form in 1985 by C. Pomerance (but with main ideas going back to M. Kraitchik in 1926), and the elliptic curve method (ECM) published in 1987 by H.W. Lenstra, Jr. ECM is suitable to find factors up to 35–40 decimal digits of large numbers. Its complexity, as conjectured theoretically, and as observed in the experiments, depends primarily on the *size* of the smallest prime factor p of the number n which we wish to factor. The complexity of the quadratic sieve method depends on the size of n, and not on its prime factors. It is still the method by which the largest numbers (not of a special form like $a^n \pm b$ where a and b are small compared to $a^n \pm b$) have been factored. The present world record is the so-called RSA–129 number, a number of 129 decimal digits. In 1977 Rivest et al. challenged the public to factor this number. They estimated that the required running time, using the best algorithms and machines available in 1977, would be 40 quadrillion ($= 10^{15}$) years. It was factored only seventeen years later, in April 1994, with a variation of the quadratic sieve method after an eight–month worldwide computing effort organized by D. Atkins, M. Graff, A.K. Lenstra, and P. Leyland. Also CWI has contributed idle workstation cycles to this result. RSA-129 turned out to be the product of two primes, one of 64 and one of 65 digits, and it is a typical example of a key used in the RSA

public-key encryption scheme. ECM and QS nicely complement each other: one usually tries ECM first in order to find factors less than 25–30 decimal digits. If one is lucky, larger factors are sometimes found: the world record is a prime factor of 42 decimal digits. In the next step QS is tried, provided that the number to be factored is small enough: popularly spoken, ECM finds smaller factors of larger numbers, QS finds larger factors of smaller numbers.

A third method, called the Number Field Sieve (NFS) and published in 1993 by J.M. Pollard, and in refined form by J.P. Buhler, H.W. Lenstra, Jr., and C. Pomerance, is expected to be more efficient for general numbers than the quadratic sieve, and it is the subject of intensive current research to find out where the cross-over point between NFS and QS lies. (See also figure 2.)

The size of the numbers which could just be factored at a given time with the available algorithms and computer technology was about 25 decimal digits in 1967, 40–50 in 1974, 70–80 in 1987, 100 in 1990, and 120–130 at present. This illustrates the rapid developments, both in algorithms and in hardware, if we realize that for the best known factoring methods the computational effort roughly *doubles* if the number to be factored grows with 2–3 decimal digits.

2.2. Primality testing

Before we are going to try to factor a number n, how do we know that n indeed is composite? Tests for compositeness are based on the 'Little Theorem' of Fermat which states that if p is a prime number, and a is a positive integer such that $\gcd(a,p) = 1$, then $a^{p-1} \equiv 1 \bmod p$. So if we find for some b with $\gcd(b,n) = 1$ that $b^{n-1} \not\equiv 1 \bmod n$, then n cannot be prime, and we can attempt to factor n. If the test yields $\equiv 1 \bmod n$, we can not be sure that n is prime since the *converse* of Fermat's Little Theorem does not hold. However, in most such cases n indeed is prime and exceptions are very rare. The simplest way to rigorously prove primality of n is to show that it has no divisors $\leq \sqrt{n}$. For small n this method works on a modern PC or workstation, but for larger n (consisting of more than 15 decimal digits, say) the number of operations becomes too large. Until 1980, the available primality tests were based on the knowledge of the prime factors of $n-1$ or $n+1$ and became impractical for numbers of more than 100 decimal digits. A breakthrough came when Adleman, Pomerance, and R. Rumely found a test that was efficient for much larger numbers. This was simplified and improved by H. Cohen, and H.W. Lenstra, Jr. An efficient implementation was written by H. Cohen and A.K. Lenstra, with the help of D.T. Winter at CWI. With this program, it was possible in 1986 to prove primality of numbers up to 300 decimal digits in a few minutes CPU-time. At present, one is able to prove primality of general numbers with more than

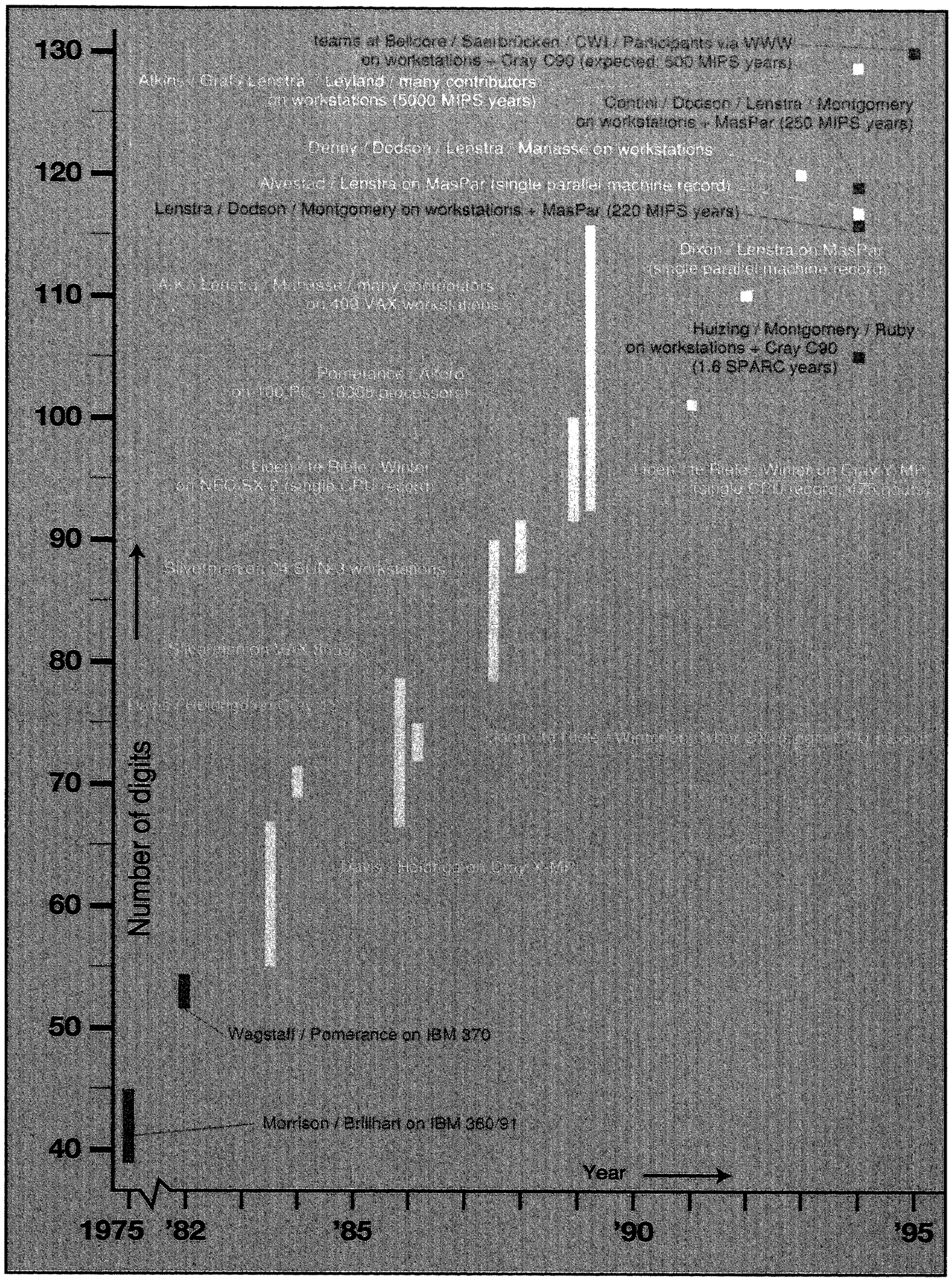

Figure 2. History of factoring records obtained with general-purpose factoring methods: Continued Fraction (blue), Quadratic Sieve (yellow), and General Number Field Sieve (red).

1000 decimal digits, thanks to algorithmic and implementational results of A. Atkin, F. Morain, W. Bosma, and M.-P. van der Hulst. For numbers of a special form, like the *Mersenne numbers* $2^p - 1$, special primality tests are known. In fact, the largest known prime number is the Mersenne prime $2^{859433} - 1$, a number of 258,716 decimal digits. It was discovered by D. Slowinski and P. Gage in 1994 with the help of the so-called Lucas-Lehmer test, which reads as follows: define the sequence $\{u_i\}$ by: $u_0 = 4$ and $u_{i+1} = u_i^2 - 2$ $(i = 0, 1, \ldots)$; then $n = 2^p - 1$ is a prime if and only if p is a prime > 2, and if n divides u_{p-2}.

2.3. Lattice basis reduction

The third problem we mention here is that of finding *small vectors in lattices.* In 1982, A.K. Lenstra, H.W. Lenstra, Jr., and L. Lovász published their so-called 'lattice basis reduction' algorithm. It computes from an arbitrary basis of a lattice in $\mathcal{R}^m$ a so-called reduced basis which has certain nice properties (its vectors are nearly orthogonal). The algorithm has many important applications in a variety of mathematical fields, like the factorization of polynomials, public-key cryptography, extracting the square-root of extremely large algebraic numbers (one crucial step in the Number Field Sieve factoring algorithm), and the disproof of the Mertens conjecture (discussed in the next section). For some applications, this algorithm in fact is a very efficient multi-dimensional continued fraction algorithm by which one is able to find simultaneous approximations of vectors of real numbers by vectors of rational numbers with the same denominator. This problem occurs frequently in number theory.

3. COMPUTATIONAL NUMBER THEORY AT CWI

A recent survey of the research in computational number theory at CWI in the past 25 years is presented in [4]. We restrict ourselves here to giving a concise description of the results obtained with respect to the Riemann hypothesis, the Mertens conjecture, and the problem of factoring large numbers. The computational number theory group at CWI presently consists of H.J.J. te Riele (project leader), W.M. Lioen and D.T. Winter (scientific programmers), and H. Boender and R.-M. Huizing (junior researchers at CWI and Leiden University). In 1993–1994, P.L. Montgomery was a visiting researcher in the group. J. van de Lune (senior researcher, retired in 1993) was the initiator of the computational work on the Riemann hypothesis, and of other projects of the group like the work on the Goldbach conjecture. Close cooperation exists with the number theory group of R. Tijdeman in Leiden.

3.1. The Riemann hypothesis

Consider the function $\zeta(s) = \sum_{n=1}^{\infty} n^{-s}$, where $s = \sigma + it$ is a complex variable. If $\sigma > 1$, then the series converges, so that $\zeta(s)$ is properly defined there. By using a technique now known as analytic continuation, Riemann showed in 1859 that there is a unique function which coincides with $\zeta(s)$ for $\sigma > 1$, and which is analytic in the *whole* complex plane, except at the point $s = 1$ (where the function has a pole of order 1). This function is known as the *Riemann zeta function*, and it plays a prominent role in prime number theory. It is known to have infinitely many complex zeros in the so-called *critical strip* $0 \leq \sigma \leq 1$, and in an eight–page paper which appeared in 1859, Riemann wrote that it is very likely that all these zeros lie *on* the line $\sigma = \frac{1}{2}$. So far, nobody has been able to (dis)prove this assertion, which is known now as the *Riemann hypothesis*.

What is the relation between the Riemann hypothesis and prime number theory? Let $\pi(x)$ denote the number of primes $\leq x$. As early as in 1792 or 1793, C.F. Gauss conjectured that the density of the prime numbers close to x is approximately equal to $1/\log x$, and that the so-called logarithmic integral

$$\mathrm{li}(x) = \int_2^x \frac{dt}{\log t} \tag{3.2}$$

is a good approximation of the function $\pi(x)$. Extensive numerical computations by A.M. Odlyzko suggest that the error in this approximation is proportional to $\sqrt{x}$: for $x = 10^{12}$, 10^{14}, 10^{16}, 10^{17}, 10^{18} we have

$$(\pi(x) - \mathrm{li}(x))/\sqrt{x} = -0.038,\ -0.031,\ -0.032,\ -0.025,\ -0.022,$$

respectively. The truth of the Riemann hypothesis implies that

$$\pi(x) = \mathrm{li}(x) + \mathcal{O}(x^{1/2} \log x) \quad \text{as} \quad x \to \infty.$$

What is known about the location of the complex zeros of $\zeta(s)$? Massive numerical computations carried out by Van de Lune, Te Riele, and Winter at CWI in 1983–1984 on a CDC Cyber 750 computer, and on a CDC Cyber 205 (one of the first vector computers in The Netherlands), have *proved* that the first 1.5×10^9 complex zeros of $\zeta(s)$ are *all simple* and lie on the line $\sigma = \frac{1}{2}$. The amount of CPU-time used was about 1000 (low-priority) hours on both machines. This extended similar computational work by R.P. Brent at the Australian National University in Canberra for the first $156,800,001$ complex zeros. Extensive computations by Odlyzko have shown later that the Riemann hypothesis holds for long sequences of consecutive zeros with rank in the neighbourhood of 10^{18}, 10^{19}, and 10^{20}. Table 1 gives the successive published records in proving that the first n complex zeros of the Riemann zeta function satisfy the Riemann hypothesis.

Investigator	n
Gram (1903)	10
Backlund (1914)	79
Hutchinson (1925)	138
Titchmarsh (1936)	1,041
Turing (1953)	1,104
Lehmer (1956)	25,000
Meller (1958)	35,337
Lehman (1966)	250,000
Rosser, Yohe, Schoenfeld (1969)	3,500,000
Brent (1979)	81,000,001
Brent, Van de Lune, Te Riele, Winter (1982)	200,000,001
Van de Lune, Te Riele, Winter (1986)	1,500,000,001

Table 1. Numerical verification of the Riemann hypothesis for the first n complex zeros.

3.2. The Mertens conjecture

The *Mertens conjecture* is a statement about the so-called Möbius function

$$\mu(n) := \begin{cases} 1, & n = 1, \\ 0, & \text{if } n \text{ is divisible by the square of a prime number,} \\ (-1)^k, & \text{if } n \text{ is the product of } k \text{ distinct primes.} \end{cases}$$

Based on numerical data concerning the function

$$M(x) = \sum_{1 \le n \le x} \mu(n),$$

F. Mertens stated in 1897 that the inequality

$$|M(x)| < \sqrt{x}, \quad x > 1,$$

is 'very probable'. This is now known as the *Mertens conjecture.*

The *size* of $M(x)$ is closely related to the location of the complex zeros of the Riemann zeta function. In fact, it is not too difficult to show that the *boundedness* of $M(x)/\sqrt{x}$ implies the truth of the Riemann hypothesis. For all values of $M(x)$ which have been computed explicitly, the Mertens conjecture is true. In 1994, Lioen and Van de Lune at CWI established that $-0.513 < M(x)/\sqrt{x} < 0.571$ for $200 < x \le 1.78 \times 10^{13}$. Their computations consumed about 400 CPU-hours on a Cray C98 super vector computer. Nevertheless, serious doubts concerning the truth of the Mertens conjecture were raised already in 1942 by A.E. Ingham, who showed that it is possible to prove the existence of certain large values of $|M(x)|/\sqrt{x}$ *without the need to explicitly compute* $M(x)$. In order to find such large values, one has to solve a so-called simultaneous inhomogeneous Diophantine approximation

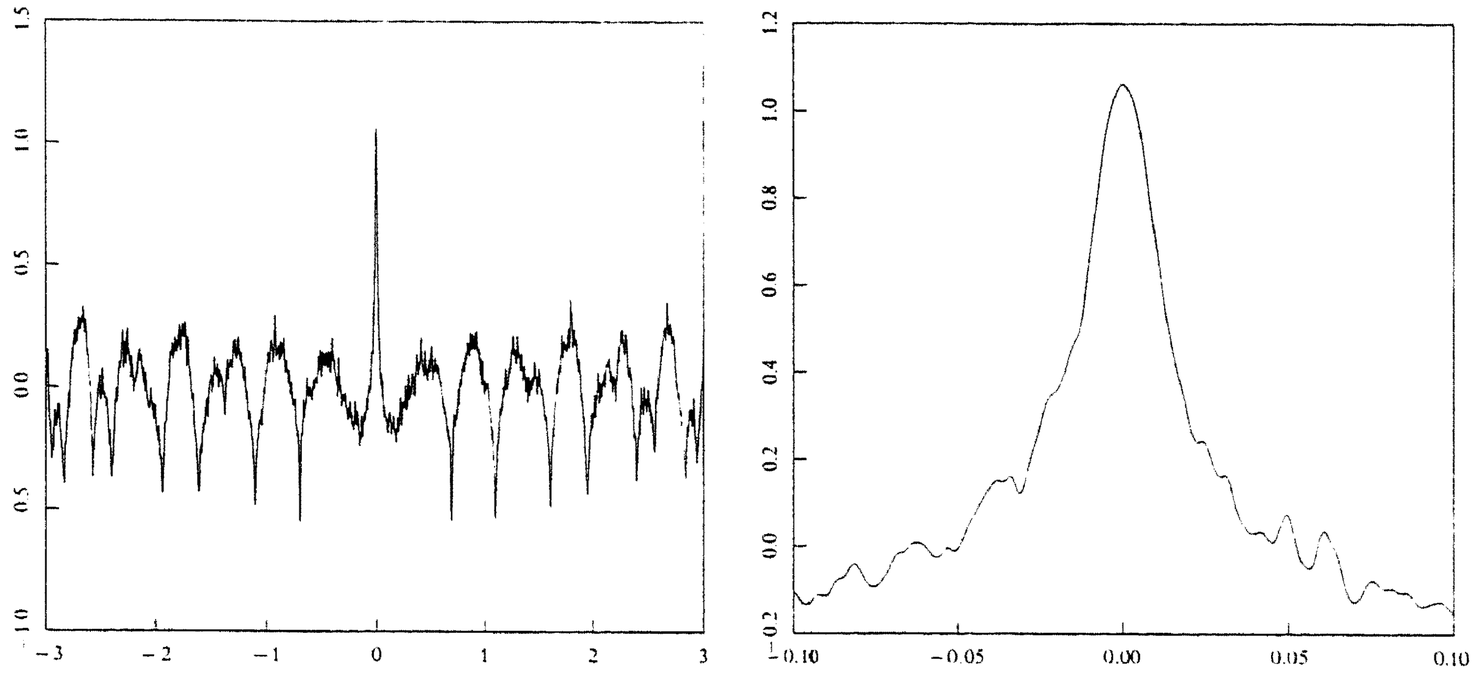

Figure 3. Graph of the function $h(y_0 + t)$ for $t \in [-3, +3]$ (left), with enlargement of its central part (right).

problem. Using the algorithm of Lenstra, Lenstra, and Lovász, mentioned in the previous section, Odlyzko and Te Riele in 1985 found a disproof of the Mertens conjecture. This required to find a value $y = y_0$ for which a certain function $h(y)$ (which we shall not give explicitly here) assumes a value > 1. The value found (see also figure 3) was:

$$y_0 = -14045\ 2896805929\ 9804679036\ 1630399781\ 1274005919\ 9978973803\ 9965960762.521505.$$

Unfortunately, this disproof is ineffective: only the *existence* of an x where $|M(x)|/\sqrt{x} > 1$ was proved. In 1987 however, J. Pintz gave an explicit (huge) upper bound, proving that $|M(x)|/\sqrt{x} > 1$ for $x \leq \exp(3.21 \times 10^{64})$. For the computation of this upper bound, Pintz used 100–digit accurate values of the first 2000 complex zeros of the Riemann zeta function, and 28–digit accurate values of the next 12950 complex zeros, as computed earlier by Te Riele for the *ineffective* disproof.

3.3. Factoring large numbers

At CWI much time and effort has been spent on the efficient implementation of the quadratic sieve method on large vector mainframes like the CDC Cyber 205, the NEC SX-2, and the Cray Y-MP and Cray C98 vector computers.

In the course of years, various new factorization records have been established by the CWI Computational Number Theory group. These, and

many other factored numbers were contributions to the so-called Cunningham Table (a table of known factors of numbers of the form $a^n \pm 1$, initiated in 1925 by A.J.C. Cunningham and H.J. Woodall) and to an extension of this table.

In Table 2 we give some figures about record factorizations found at CWI on vector computers. All results were obtained on *one* processor of the vector computer listed. On the Cray Y-MP we *could* have used four CPUs, thus reducing the sieving time by a factor of about four, since the most time-consuming steps of the quadratic sieve algorithm are almost perfectly parallelizable.

year	machine	size of numbers (decimals)	sieving time (hours)	Gaussian elim. time (seconds)	approximate order of sparse system
1986	Cyber 205	72	4.3	21	6,070
		75	12.2	37	7,400
1988	NEC SX-2	87	30	200	18,800
		92	95	700	24,300
1991	Cray Y-MP	101	475	1800	50,200

Table 2. Record factorizations with QS on vector (super)computers.

The latest records were obtained in the summer of 1994 with the help of the Cray C98 at SARA (The Academic Computing Centre Amsterdam), and many workstations in a collaboration between Oregon State University and CWI: a 162-digit Cunningham number was factored with the 'Special Number Field Sieve' (SNFS, for which the number n to be factored has the form $n = a^m \pm b$, a and b being small compared to n), and a 105-digit number was factored with the 'General Number Field Sieve' (GNFS, for which no special form of n is known). One month after the latter result was obtained, A.K. Lenstra, B. Dodson, and Montgomery cracked a 116-digit partition number with GNFS. On November 26, 1994 S. Contini, Dodson, A.K. Lenstra, and Montgomery completed the factorization of a 119-digit cofactor of the 123-digit partition number $p(13171)$ into two primes of 52 and 67 digits using GNFS. From the time they used (about 250 mips years) they estimate that this is about 2.5 times less than what they would need to factor a number of comparable size with the quadratic sieve method.

Montgomery and Huizing factored several other numbers with SNFS (of 98, 99, 106, 119, 123, 135, and 137 decimal digits) including some *more* and *most* wanted Cunningham numbers (i.e. difficult numbers in the Cun-

ningham table, not yet factored) using a new algorithm of Montgomery for computing the square root of the product of many algebraic numbers, and his new iterative block Lanczos algorithm for finding dependencies in large sparse matrices over GF(2). Huizing also factored 87-, 97-, and 107-digit numbers with GNFS.

Currently, most factorization research at CWI aims at contributing to the Cunningham tables. In the first update to the extended table, issued in September 1994, all the composite numbers with less than 86 decimal digits were completed. This bound has been raised in May 1995 to 90 decimal digits.

References

1. H. Cohen (1993). *A Course in Computational Algebraic Number Theory*, Volume 138 of *Graduate Texts in Mathematics*, Springer–Verlag, Berlin.
2. P.L. Montgomery (1994). A survey of modern integer factorization algorithms. *CWI Quarterly, 7(4).*
3. M. Pohst, H. Zassenhaus (1989). Algorithmic algebraic number theory. *Encyclopedia of Mathematics and Applications*, Cambridge University Press, Cambridge.
4. H.J.J. te Riele, J. van de Lune (1994). Computational number theory at CWI in 1970 – 1994. *CWI Quarterly, 7(4).*
5. H.C. Williams, J.O. Shallit (1994). Factoring integers before computers. W. Gautschi (ed.). *Mathematics of Computation 1943–1993: a Half-Century of Computational Mathematics*, 481-531. Proceedings of Symposia in Applied Mathematics, American Mathematical Society.