BULLETIN (New Series) OF THE
AMERICAN MATHEMATICAL SOCIETY
Volume 39, Number 3, Pages 449–454
S 0273-0979(02)00947-3
Article electronically published on April 15, 2002

Prime numbers: A computational perspective, by Richard Crandall and Carl Pomerance, Springer, New York, 2001, xv + 545 pp., $49.95, ISBN 0-387-94777-9

The twin problems of determining if a given positive integer n is prime and, if it isn't, finding a nontrivial factorization of it are so fundamental that they demand serious theoretical attention. They are also peculiarly rewarding; even today one can imagine the satisfaction Euler must have felt when he discovered that

$$2^{32} + 1 = 4294967297 = (641)(6700417),$$

disproving Fermat's guess that the numbers $2^{2^n} + 1$ are prime in general. Perhaps this is because these problems cater to the competitive spirit; unlike many mathematical problems they are rather clearly graded by difficulty – my prime beats yours if it has more digits. Whatever the reason, mathematicians from Fermat's time to the present have been bringing the best techniques available from all areas of mathematics to extend the range of numbers that they can factor or prove prime. The discovery in the 1970's of public-key cryptography and its relation to factoring and primality testing has added the profit motive as a driving force behind understanding factoring, further stimulating work on these problems.

The elementary school method for solving both the primality proving problem and the factoring problem is to systematically divide a candidate number N by the numbers less than N. Either a factor turns up, or, by ruling out all potential factors less than $\sqrt{N}$, one proves N prime. This beautiful method is, unfortunately, completely useless for attacking the general problem once the wider world of large numbers comes under study. For example, suppose that it is possible to do 10^{10} trial divisions per second. To determine if an 80 digit number is prime would require about 10^{40} trial divisions, requiring about 10^{30} seconds. This would take more than 10^{20} years, or far longer than the time since the big bang.

The most powerful methods today for factoring are Pollard's Number Field Sieve (NFS) and Lenstra's Elliptic Curve Method (ECM), while for primality proving the best methods, such as the Atkin-Morain method, are also derived from elliptic curves. Using these methods, the current recordholders have been able to achieve primality proofs of numbers of thousands of digits and factorizations of numbers of sizes approaching 200 digits.

Describing these algorithms is a challenge for any author, because the general subject of factoring and primality proving includes a wide spectrum of problems. At one end of the spectrum, understanding what to expect from the running time of these algorithms, and optimizing the parameter choices they involve, requires deep techniques in analytic number theory. At the other end of the spectrum, implementing the algorithms so as to get practical results at the limit of what is possible requires expertise in the more delicate aspects of computer programming.

The book under review, *Prime numbers: A computational perspective*, by Richard Crandall and Carl Pomerance, is a survey of the most important modern methods for primality proving and factoring that represents a collaboration between two

2000 *Mathematics Subject Classification.* Primary 11Y40.

authors, one theoretically and one computationally inclined (Pomerance and Crandall respectively). Thanks to this collaboration, the book represents a satisfying balance between theory and practice. The authors include detailed information on implementing these algorithms, including psuedocode and numerical examples that can be used for testing. They also include a convincing discussion of the underlying number theory, often quoting results and explaining the role they play in the algorithms, rather than proving theorems in analytic number theory. Not exactly a textbook in the traditional sense, this book's informal style, open-ended problems, and myriad references capture the freewheeling nature of work in this area. The vast assortment of problems is an outstanding source of ideas for experimental projects for undergraduate research and also contains many suggestions for possible directions for new research at the frontiers of the subject. This book is an excellent resource for anyone who wants to understand these algorithms, learn how to implement them, and make them go fast. It's also a lot of fun to read.

Prime numbers begins with an eclectic survey of problems, known results, and "curiosities" about prime numbers. The topics include up-to-date status reports on such chestnuts as the twin prime conjecture and the Goldbach conjecture. There is also a survey of results related to the prime number theorem and the Riemann hypothesis. To get a sense of the flavor of the book, look more closely at how one of these topics is presented. Section 1.3.1 discusses Mersenne primes (primes of the form $M_q = 2^q - 1$). In this brief section Crandall and Pomerance give proofs of some elementary results on such numbers, such as the relation to even perfect numbers due to Euler. They provide a table of known Mersenne primes as of November 2000, where we see that $M_{6972953}$ was the largest known proven prime at that time. They discuss a range of conjectures about numbers M_q and derive a heuristic estimate on the chance of M_q being prime that relies on Mertens' theorem; the estimate is that M_q is prime with probability $e^\gamma \ln(q)/q \ln(2)$ where γ is Euler's constant. Finally they offer Problem 1.29, which begins by suggesting some computational problems such as ruling out odd perfect numbers below some bound, finding an odd number n such that $\sigma(n) > 2n$ where σ is the sum of divisors function, and studying numerically $\sigma(n) - n$, and ends by posing an unsolved problem related to $\sigma(n) - n$ with references to related work of H. Lenstra and P. Erdos. This is a truly "wide-ranging" treatment.

Chapter 2 presents the algorithms that lie at the foundation of the techniques for primality proving and factoring treated later in the text. These include methods for computing greatest common divisors, effective versions of the Chinese remainder theorem, and Jacobi symbols. The even more fundamental algorithms concerned with arithmetic (meaning elementary school arithmetic, like multiplication and division) are treated much later, in considerable detail, in Chapter 9. The computational perspective of the book is illustrated by the treatment of quadratic reciprocity, which the authors state without proof. They then present an algorithm for computing the Jacobi symbol using reciprocity in pseudocode and point out its complexity. A proof of quadratic reciprocity, based on some computations of Gauss sums, is posed as Problem 2.24.

The serious study of primality testing and proving begins in Chapter III of *Prime numbers*, with an extensive discussion of pseudoprime tests. The simplest of these tests is the Fermat test: Suppose that N and a are positive integers, and $a^N \not\equiv a \pmod{N}$; then N is composite. It is very easy (in comparison with factoring) to compute $a^N \bmod N$ using for example the algorithms of Chapters 2

and 9 of *Prime numbers*, and so testing the values $a^N \bmod N$ for several a is a good way to test a number N for compositeness. In practice most composite numbers are detected immediately by Fermat tests. However, there are numbers, called Carmichael numbers, that are composite yet have the property that $a^N \equiv a \pmod N$ for all a. The smallest such number is 561, and Alford, Granville, and Pomerance proved in 1994 that there are infinitely many more.

The idea behind the Fermat test admits considerable elaboration. The first such is the strong pseudoprime test, which combines the Fermat test with the observation that, if N is prime, and $x^2 \equiv 1 \pmod N$, then $x \equiv \pm 1 \pmod N$. To apply the strong pseudoprime test, write $N - 1 = 2^r M$ with M odd and compute the sequence of numbers $a^M, a^{2M}, \ldots, a^{2^r M}$. If N is prime and a is not divisible by N, then either $a^M \equiv 1 \pmod N$ or we must see a 1 in this sequence preceeded by a -1. There are no Carmichael-type numbers for the strong pseudoprime test; for any composite N there is some a so that the strong pseudoprime test will detect this fact. Such an a is called a "witness". Witnesses are common (at least 3/4 of the numbers are witnesses; see Theorem 3.4.4), but the smallest witness could be large (see Section 3.4.1).

While discussions of the Fermat and strong pseudoprime test are common in number theory texts nowadays, the treatment here is notable for its thoroughness. Most elementary treatments of pseudoprime tests avoid completely the issue of the distribution of witnesses, which has led to considerable interesting work in analytic number theory, but *Prime numbers* summarizes the state of work on witnesses. In addition, *Prime numbers* goes much further than general texts do into the entire theory of pseudoprime testing, making the point that such tests, which go under many names, are collectively based on the fact that the multiplicative group of a finite field is cyclic. For example, the Lucas tests are based on the fact that if N is prime and $x^2 + ax + b$ is irreducible mod N (in other words if $a^2 - 4b$ is a square mod N which can be tested using quadratic reciprocity), then in the finite field $(Z/NZ[x])/(x^2 + ax + b)$ we must have $x^{N+1} = b$.

As is the case throughout the book, the authors pay full attention to the practical problems involved in implementing such tests, providing pseudocode and extensive discussion, and offer many open-ended problems based on their discussion. For example, problem 3.43 asks, for each k, for the first number n_k with least witness k. The authors report in one of the problems for this section that $n_{23} = 341550071728321$.

Chapter 4 turns from detecting composite numbers to proving numbers prime. Much of primality proving is based on variants of Pocklington's theorem; the underlying theoretical idea is that if $(Z/NZ)^*$ is cyclic of order $N - 1$, then N is prime. More explicitly, suppose that $N - 1 = FR$, with $F \geq R$. Suppose further that we know the complete factorization of F and that we can find a such that

$$a^{N-1} \equiv 1 \pmod N \text{ and } \gcd(a^{(N-1)/q} - 1, N) = 1 \text{ for each prime } q|F.$$

Then N is prime. Applying Pocklington's theorem requires a partial factorization of $N - 1$ into known primes. Many of the refinements of this approach are based on getting by with an even more partial factorization (i.e. weakening the condition $F \geq R$) or replacing $(Z/NZ)^*$ by the units in $(Z/NZ)[x]/(x^2 + ax + b)$, or by a combination of both techniques. The elliptic curve methods that are treated in Chapter 7 are further elaborations of this idea.

Chapter 4 concludes with a discussion of the Gauss and Jacobi sum tests of Adelman, Pomerance, and Rumely. This method is important both theoretically and practically: theoretically because its running time is rigorously proved to be very nearly polynomial time in the number of digits in N, and practically because it is effective at proving specific large numbers prime. *Prime numbers* contains a complete description of the algorithm and a proof that it works, along with a precise statement of the result from analytic number theory needed to establish the running time. However this algorithm is considerably more sophisticated than the other algorithms discussed earlier, and the authors don't provide any numerical examples; as a result the presentation of this particular algorithm is less satisfying (and less useful to someone planning an implementation) than that of many others in the book.

Chapters 5 and 6 treat factoring algorithms, with Chapter 5 devoted to exponential time algorithms (meaning algorithms whose running time is $O(N^k)$ for some k) and Chapter 6 to the Number Field Sieve (NFS) and its earlier relative, the still powerful Quadratic Sieve (QS). QS and NFS, as well as other related algorithms such as the Brillhart-Morrison continued fraction algorithm, are based on the simple idea of locating integers X and Y with the property that $X^2 \equiv Y^2 \pmod{N}$. Given such a pair of integers, one can hope that $\gcd(X - Y, N)$ is a non-trivial factor of N. If one can produce enough random pairs of congruent squares, the odds of obtaining a factorization in this way becomes very high.

For the quadratic sieve, one creates a "factor base" of primes up to some bound B. Then, using a technique known as "sieving", one generates a series of squares modulo N that factor completely using only primes up to B. Each such successful factorization yields a *relation*

$$a^2 \equiv {p_1}^{e_1} \cdots {p_k}^{e_k} \pmod{N}$$

where $p_1, \ldots, p_k$ are the primes up to N. Given enough such factorizations (and for the largest factorizations, collecting these relations may require thousands of computers using their spare cycles for a year) one can use linear algebra on the exponent vectors to construct a multiplicative combination of the a's on the left hand side which is congruent to a product of even powers of the p_i. This gives an equation $X^2 \equiv Y^2 \pmod{N}$ and a chance at a factorization. The number field sieve extends this idea but uses instead factorizations in a ring of algebraic integers.

The asymptotic running time of these algorithms depends on how likely numbers of a certain size are to factor into primes less than B. Here one faces the basic tradeoff that increasing B makes this factorization more likely, but also increases the size of the linear algebra problem so that more relations are required. Using results on the distribution of smooth numbers and making some highly unprovable heuristic assumptions, especially in the case of the number field sieve, one can show that these algorithms require subexponential time. For the quadratic sieve, the accepted asymptotic estimate is on the order of $\exp((1+o(1))\sqrt{\ln N \ln\ln N})$, while for the number field sieve it is

$$\text{Time to factor N using NFS } = \exp\left((64/9)^{1/3} + o(1)\right)(\ln N)^{1/3}(\ln\ln N)^{2/3}).$$

Prime numbers includes many insights into the kinds of optimizations that the pros use to efficiently implement NFS and QS for large factorizations. One particularly useful feature is that, in the problem section, the authors provide some worked examples, with numerics, that can be used to check a home-made implementation.

With this book in hand, one can imagine a determined undergraduate with a background in elementary number theory and some strong programming ability coming up with a solid QS implementation in a reasonable time.

Crandall and Pomerance also outline the heuristics supporting the running time estimates quoted above. These heuristics require understanding the function

$$\psi(x, y) = \{\text{Number of } a \leq x \text{ with all prime factors } p \text{ of } a \text{ smaller than } y\}$$

that controls how likely the numbers a^2 generated by the sieving algorithms are to factor into primes less than B. The study of this function (psixyology) is an interesting branch of analytic number theory that contributes directly to the understanding of factoring algorithms, but reflecting the book's bias towards computation as opposed to analytic number theory, the authors content themselves with quoting the results on $\psi(x, y)$ that they need.

Chapter 7 covers applications of elliptic curves to factoring and primality proving and discusses the Schoof algorithm for determining the number of points on an elliptic curve mod p. The factoring and primality proving applications of elliptic curves are natural modifications of the methods of Chapter 5 based on finite fields. For example, the $p - 1$ method discussed in Chapter 5 uses the fact that, if N is composite, the multiplicative group $(Z/NZ)^*$ splits as $(Z/AZ)^* \times (Z/BZ)^*$; we could hope to find a power M and an integer mod N so that $a^M \equiv 1 \pmod{A}$ but $a^M \not\equiv 1 \pmod{B}$; then $(a^M - 1, N)$ would detect a factor of N. The elliptic curve method relies similarly on the idea that the group of sections $E(Z/NZ)$ on an elliptic curve over Z/NZ splits as $E(Z/AZ) \times E(Z/BZ)$, and we could hope to find a point P on E and a multiple b so that $[b]P = 0$ in $E(Z/AZ)$ but not in $E(Z/BZ)$. This phenomenon can easily be detected in the course of the computation of the multiples of P and reveals a factor of N.

The influence of a computational perspective shows up rather strikingly in the elliptic curve chapter. The authors quote Hasse's famous theorem that the number of points on E mod p is within $2\sqrt{p}$ of $p + 1$, and they spend a significant amount of time analyzing the formulae used to compute multiples of points on E, trying to reduce the number of time consuming operations like inversion modulo N. As with the QS and NFS, they provide pseudocode and numerical examples which will be very helpful to anyone setting out to construct an ECM implementation.

The other topics of Chapter 7 are primality proving using elliptic curves and point counting on elliptic curves via Schoof's algorithms. This is one of the few places where these topics are treated in a reasonably comprehensive and yet accessible way.

Chapter 8 introduces some applications of prime numbers to "applied" problems. The cryptographic applications are widely known and widely discussed, and they are treated here with dispatch. More interesting, and again a good source of projects, are the sections on Monte Carlo methods, quantum computation, and random number generators.

The book concludes with a survey of methods for carrying out multiple precision integer arithmetic and fast polynomial arithmetic. This topic tends to be neglected in textbook treatments of number theoretical algorithms, but it is absolutely central to the efficient implementation of any of the algorithms discussed earlier in the book. The authors discuss many of the different flavors of the Fast Fourier Transform, with pseudocode and numerical examples, along with references to specific applications in factoring. This chapter includes 21 pages of problems for further investigation.

For a variety of reasons, *Prime numbers: A computational perspective* brings to mind Knuth's *Art of computer programming*, especially Volume II of that work. Some of this resemblance is superficial – for example, both books include algorithms in pseudocode, and both include problems of widely varying difficulty. Indeed, Crandall and Pomerance might have helped their readers a bit by including some form of Knuth's system of rating problem difficulty. Some of the resemblance is a feature of subject matter; many of the topics treated by Crandall and Pomerance figure also in Knuth's work, and indeed on many of these topics, such as fast multiplication and the euclidean algorithm, Knuth has a great many relevant things to say. But most importantly, *Prime numbers*, like Knuth's work, teaches the unity of mathematics, and the inherently mathematical nature of efficient computation, by freely drawing on a wide range of mathematical techniques to illustrate computational problems from many points of view and by emphasizing the mathematical ideas which make efficient computation possible. It's rare to say this of a math book, but open *Prime numbers* to a random page and it's hard to put down. Crandall and Pomerance have written a terrific book.

Jeremy Teitelbaum
University of Illinois at Chicago
E-mail address: jeremy@uic.edu