Recent Developments in Primality Testing

Carl Pomerance*

Given a natural number n, how quickly can you tell if it is prime or composite? This basic and centuries old problem has seen a great profusion of activity within the past 5 or 6 years. If anything in mathematics is "elementary", certainly one would think that this problem is. Thus it was very surprising when a proof was announced that linked the problem to the Extended Riemann Hypothesis. But there were other major developments in primality testing as well. Two fast running probabilistic algorithms served to focus philosophical concern on the nature of mathematical proof. A number was proved to be prime that was larger than the square of the astronomically large champion of only six months earlier. An application to cryptography of the ability to find large primes and the inability to factor quickly made headlines in newspapers and called into question the role of the U.S. government in the usually insulated American mathematical community. On other fronts, researchers steadily added to the storehouse of practical methods for primality testing, so that now it is not unusual for an 80 digit prime of no special form to be supplied with a proof of its primality. And late in 1980, a new primality testing algorithm was announced that asymptotically runs in nearly polynomial time, is of theoretical interest because of its use of algebraic number theory, specifically the higher power reciprocity laws, and perhaps has the potenital for real implementation on numbers of several hundred decimal digits.

In what follows we shall briefly describe these exciting developments. However the cryptography application has already been amply discussed in the Mathematical Intelligencer and elsewhere (see [12], [29], [31]), so we shall say no more about it here. In addition we would like to call attention to the excellently written survey articles on primality testing by Lenstra [17] and Williams [35].

Some Simple Algorithms

Everyone knows that the prime or composite nature of a given number n can be determined by trial divisions of n by the numbers (or prime numbers) up to $\sqrt{n}$. This algorithm has the advantage of not only providing a proof of primality for prime n, but of discovering a non-trivial factorization for composite n. The main drawback of the trial division algorithm is that it takes too long if n has no small prime factors. For example, say we use the trial division algorithm on a computer that can do one million trial divisions per second to determine if a given n is prime or composite. If n is a prime near 10^{40}, the running time will be about one million years, while if n is a prime near 10^{50}, the age of the universe would not suffice!

The trial division algorithm is an example of an exponential time algorithm. If the input n has k decimal digits, then the number of computational bits involved in running the algorithm could be as many as $\sqrt{n} > (\sqrt{10})^{k-1}$. Since $(\sqrt{10})^{k-1}$ is an exponential function of k, the length of the input, we say the algorithm runs in exponential time. A major unsolved problem in the theory of algorithms and in number theory is whether there is a polynomial time primality testing algorithm. That is, the running time is $O(k^c) = O(\log^c n)$ where c is a constant and n is the number to be tested for primality. (Recall that we say $f(x) = O(g(x))$ if there is a constant C such that $|f(x)| \leqslant Cg(x)$ for all values of x in question.)

We now describe an algorithm with the property that the average time spent on any one number is very small. It is the familiar sieve of Eratosthenes. Here one starts with a list of all numbers from 2 to some point x (there are variations where we require only that the list be in arithmetic progression or the consecutive values of some polynomial). At each stage, one circles the first unmarked number n, striking out every n-th number thereafter, unless $n > \sqrt{x}$, in which case one circles every remaining unmarked number and stops. The circled numbers are a complete list of primes up to x. This algorithm is excellent for constructing such a list and/or factor table of least prime factors of composite numbers. Moreover, the average time spent on any one number is very small -- it is $O(\log\log n)$. However one is obviously limited in the execution of this algorithm by the amount of storage space in the computer used and by the length of the print-out. Even though for any one n the time spent considering it is very small, one must consider all of a large set of integers or none of them.

Both the trial division algorithm and the sieve of Eratosthenes not only can prove primality but also can factor. And factoring is hard. It has been estimated conservatively in [29] that to factor certain composite n with 75 de-

* Supported in part by a grant from the National Science Foundation.

cimal digits, using the fastest factoring algorithm known (which is neither of the above) on an imaginary computer faster than any now in existence would take about 15 weeks. For certain numbers with 100 digits it could take a life time.

Perhaps one could gain time by not asking for a factorization of composite n. That is, if our only goal is to determine whether n is prime or composite and not to factor it in the latter case, perhaps there are speedier algorithms. Indeed there are, as we shall see below.

The Converse of Fermat's Little Theorem

The "little theorem" of Fermat asserts that if $(b, n) = 1$ and n is prime, then

$$b^{n-1} \equiv 1 \bmod n. \tag{1}$$

This fact, at least for the case $b = 2$, was already known to the fifth century B.C. Chinese (see Dickson [5], pp. 59, 91). However, the Chinese (and Leibniz) also asserted the converse; that is, if (1) holds for $b = 2$, then n is prime. Wouldn't it be nice if this were true! To see if a large n is prime one would only have to compute the residue of 2^{n-1} mod n and see if it is 1. Moreover, it is a simple computation to compute the residue of b^{n-1} mod n. Indeed, one writes $n - 1$ in binary, so that $n - 1 = B_0 + B_1 \cdot 2 + \ldots + B_k \cdot 2^k$ where each $B_i = 0$ or 1; one finds the residues of the b^{2^i} mod n, $0 \leqslant i \leqslant k$, by successively squaring; and then, corresponding to those i with $B_i = 1$, one successively multiplies together these residues, reducing mod n after each multiplication. This procedure is a polynomial time algorithm, taking $O(\log^3 n)$ computational bits. Such a computation can be practically accomplished in minutes on an unextraordinary computer even if n has thousands of digits. Recall that the trial division algorithm would require more than the age of the universe to establish primality for a number of just 50 digits.

Unfortunately, the converse of Fermat's little theorem is not true. For each $b \neq 0$ there are infinitely many composite n for which (1) holds. For example, (1) holds for $b = 2$ and $n = 341 = 11 \cdot 31$. This can be mentally verified since clearly $2^5 \equiv 1$ mod 31 and by Fermat's little theorem, $2^{10} \equiv 1$ mod 11. A composite n for which (1) holds is called a *pseudoprime to the base b*.

For a fixed value of b, it is an observed fact that the pseudoprimes to the base b are very rare, much rarer than primes. For example, below 10^{10} there are 455052512 primes, but only 14884 pseudoprimes to the base 2 (see [14] and [25]). The observed rarity of pseudoprimes coupled with the ease of numerically checking (1) indicate that perhaps we should not give up on (1) as a primality test: we just need some further test to weed out the pseudoprimes. The conjectured rarity of pseudoprimes compared with primes was proved in 1950 by Erdös [7] (also see [24]).

Connections with the Extended Riemann Hypothesis

Because for each fixed b there are composite n for which (1) holds, one possibility for a primality test might be to vary b. But again there are composite n that masquerade as primes. For example $n = 561 = 3 \cdot 11 \cdot 17$ satisfies (1) for *every* b with $(b, 561) = 1$. (This too can be mentally verified by noting that 2, 10, and 16 each divide 560.) Such values of n which are pseudoprimes to every base b with $(b, n) = 1$ are known as Carmichael numbers. It is widely believed there are infinitely many of them, but this has not been proved.

If $n > 2$ is prime, then 1 has just two square-roots in the integers mod n: 1 and -1. If n is the product of k distinct odd primes, then by the Chinese Remainder Theorem 1 has 2^k square-roots in the integers mod n. This fact suggests a strengthening of (1), since if (1) holds for the odd number n, then $b^{(n-1)/2}$ is a square-root of 1 in the integers mod n. If it is not congruent to ± 1, then n is composite. For example, even though $5^{560} \equiv 1$ mod 561, we have $5^{280} \equiv 67$ mod 561, so 561 is recognized as composite.

On input of an odd integer n, we can perform the following test. Write $n - 1 = 2^s m$ where m is odd. If either

$$b^m \equiv 1 \bmod n \quad \text{or} \quad b^{2^i m} \equiv -1 \bmod n$$

for some i, $0 \leqslant i \leqslant s - 1$, then n passes the test. If $n \nmid b$ and n fails the test, then n is composite. This test was first proposed (in a slightly different form) by Miller [20] (cf. Williams [35], § 17). If an odd composite n passes Miller's test, we shall call n a *strong pseudoprime to the base b*. A strong pseudoprime test takes no more time than before, that is, $O(\log^3 n)$ bit operations. Again, unfortunately, for every $b \neq 0$ there are infinitely many strong pseudoprimes to the base b (see [23], [25]).

Empirically, however, there are very few integers that are simultaneously strong pseudoprimes to several predetermined bases. For example, in [25] it is announced that there is only one integer $n < 25 \cdot 10^9$ which is simultaneously a strong pseudoprime to the bases 2, 3, 5, and 7. This integer is

$$3215031751 = 151 \cdot 751 \cdot 28351,$$

and it is not a strong pseudoprime to the base 11. Thus a programmable calculator or small computer contains within it a table of primes to $25 \cdot 10^9$. Simply perform the 5 mentioned strong pseudoprime tests. If n passes and $n < 25 \cdot 10^9$, then n is prime. (Or one could perform only the first 4 tests and then check that your number isn't the one exception.)

In fact there is no composite n which is a strong pseudoprime to every base b with $(b, n) = 1$; there is no analogy to the Carmichael numbers. To see this, suppose p_1 and p_2

are distinct odd primes dividing n (the prime power case is easy to see by a separate argument.) If n is a strong pseudoprime to the base b, then the multiplicative orders of b modulo p_1 and p_2 must be divisible by precisely the same power of 2. (That's easy!) Now suppose that 2^{s_i} is the power of 2 which exactly divides $p_i - 1$, and assume that $s_1 \leqslant s_2$. Recall that the Legendre symbol (b/p) is 1 or -1, depending on whether b is or is not a square mod p. Of course, b is *not* a square mod p_i precisely when 2^{s_i} divides its order mod p_i. Thus if $(b/p_1) = 1$ and $(b/p_2) = -1$, then n can *not* be a strong pseudoprime to the base b. Furthermore, the Chinese Remainder Theorem provides examples of such b.

In fact, we can define (see Lenstra [16])

$$\lambda(b) = \begin{cases} \left(\dfrac{b}{p_2}\right) & \text{if } s_1 < s_2 \\[2ex] \left(\dfrac{b}{p_1}\right)\left(\dfrac{b}{p_2}\right) & \text{if } s_1 = s_2, \end{cases}$$

and check that n is not a strong pseudoprime base b whenever $\lambda(b) = -1$. Since λ is a homomorphism from the multiplicative group of reduced residues mod n onto the cyclic group of order 2, at least half the integers between 1 and n will tell us that n is composite!

Miller [20] exploits this situation as follows (actually, Miller's argument is a little different). Recall that a character mod n is a homomorphism χ from the multiplicative group of reduced residues mod n to the non-zero complex numbers. Ankeny [2] has shown that if the Extended Riemann Hypothesis (ERH) is assumed, then there is an absolute constant c such that if χ is any non-constant character mod n, then there is some positive integer $b < c \log^2 n$ with $(b, n) = 1$ and $\chi(b) \neq 1$. Since λ, defined above, is a non-constant character mod n, it follows from Ankeny's conditional result that there is a small b such that n is not a strong pseudoprime to the base b.

As we mentioned above a strong pseudoprime test for n may be accomplished in $O(\log^3 n)$ bit operations, and thus Miller proposes that we so test n for each base $b < c \log^2 n$, for a total of $O(\log^5 n)$ bit operations. If n is composite (and the ERH is true), then n will fail at least one of the strong pseudoprime tests. If n passes all of these tests (and the ERH is true), then n is prime. Thus if the ERH is true, the issue of whether n is prime or composite can be decided in polynomial time! (There is a special argument to take care of the case when n is a prime power -- the above analysis fails here.)

The catch, of course, is the reliance on the ERH. Although widely (but not universally) believed to be true, the ERH has only the status of a conjecture. Briefly, the ERH includes the ordinary Riemann Hypothesis plus the statement: if χ is a non-constant character mod n, then the analytic function defined by the series

$$L(s,\chi) = \sum_{\substack{a=1 \\ (a,n)=1}}^{\infty} \frac{\chi(a)}{a^s}$$

in the half plane Re $s > 0$, has all its zeroes on the line Re $s = 1/2$.

A similar speedy, but conditional test was recently proposed by Selfridge and Weinberger (unpublished, but see [35], § 21). They replace the strong pseudoprime tests with the simpler test, $b^{(n-1)/2} \equiv \pm 1 \bmod n$.

Probabilistic Compositeness Tests

A probabilistic algorithm is one for which there is a possibility of a very long, perhaps infinite running time. In practice, the algorithm will of course have a special instruction to halt if some time limit or pre-set number of steps is exceeded. However, if the algorithm terminates of its own accord before this time limit, the conclusion is true — the element of chance only concerns the running time. There is at least one other definition of a probabilistic algorithm (for example, see Rabin [27], p. 24), but for the sake of clarity we shall use the term in the above sense solely.

In the last section we noted that no composite n can be a strong pseudoprime to every base b with $(b, n) = 1$. In fact, Rabin [28] has proved that no composite n can be a strong pseudoprime for $n/4$ choices of a base b in $[1, n-1]$. (The argument we gave in the last section can be adapted to show this result if n is divisible by at least 3 distinct primes – special arguments are then constructed for the remaining cases.) Thus, if b is randomly chosen in the interval $[1, n-1]$, the probability that the composite number n will pass a strong pseudoprime test to the base b is at most 1/4. Of course the actual probability is a function of n, but 1/4 is a universal upper bound for all n. Thus Rabin proposes the following probabilistic algorithm: on input n, perform strong pseudoprime tests on n for randomly chosen bases in $[1, n-1]$, stopping as soon as n fails a test. Of course, if n is prime, the algorithm will never stop (unless it is instructed to do so after $n/4$ distinct bases are chosen). But if n is composite, the algorithm will usually stop after at most a few tests. In fact, for composite n the probability that the algorithm will terminate after at most k tests is at least $1 - 4^{-k}$. If we predetermine k, say $k = 100$, and agree to run at most k tests, then the number of bit operations is $O(\log^3 n)$. The Rabin test for a fixed k cannot prove that a prime n is prime -- it should not be referred to as a primality test, but as a compositeness test.

However, Chaitin and Schwartz [39] have shown that if a truly random sequence is used in connection with this algorithm, it will become a polynomial time deterministic primality test! The catch is, unfortunately, that it is a consequence of the Gödel theory that it is impossible to prove that any particular sequence is truly random, although most sequences are.

Another probabilistic compositeness algorithm, where a different pseudoprime test (originally discovered by Lehmer [15]) replaces the strong pseudoprime test, was proposed earlier by Solovay and Strassen [33]. The two algorithms are very similar, but Rabin's is slightly easier to explain and so is the one presented here.

These probabilistic algorithms have served to focus an interesting philosophical question on the nature of mathematical truth (see Kolata [11] and DeMillo, Lipton, and Perlis [4]). The argument goes as follows. It is only an illusion or wishful thinking that there is absolute certainty in mathematics. It is commonly accepted that the natural "laws" in the other sciences are not immutable, but subject to revision as scientists gain deeper understanding. Witness the developments in physics during this century. In fact, mathematics is no different. Mathematicians know many instances of counter-examples for so-called theorems (often, their own). Of course, occasionally a proof is just plain wrong and cannot be patched up. But often the exhibition of a counter-example causes us to tighten definitions or to broaden the theorem to include new cases. This dialectic between proofs and counter-examples is the life blood of mathematics (see Lakatos [13]). Because of this uncertainty in every human endeavor, including mathematics, we might assign probabilities of truth to mathematical theorems. A simple result that has borne the test of time, such as the equation $2 + 2 = 4$, should be given a probability of truth that is extremely close to 1. A more complicated result, such as the Fundamental Theorem of Algebra, should be given a slightly lower probability of truth. A new and complicated result, such as the recent classification of finite simple groups, should be given an even lower probability. Of course, this "moral arithmetic" is somewhat arbitrary and seems to imply the questionable view that there is absolute truth out there somewhere.

Say we accept the above argument, especially the "moral arthmetic" part. And say n passes 100 randomly chosen strong pseudoprime tests. Then the probability that n is prime is at least $1 - 4^{-100} - p$, where p is the probability that there is some important error in the Rabin analysis, or that the computer made an error in implementation, or that the computer was programmed incorrectly. Say we now employ the trial division algorithm on n and conclude that n is prime. Then the probability that n is prime is at least $1 - p'$, where p' is analogous to p. Since the trial division algorithm requires much more computation than 100 strong pseudoprime tests (if n is large), we might guess that $p' > p + 4^{-100}$. Thus, by this argument the Rabin test is a better test for primality than trial division in that we can be more sure of the outcome.

On the other side we might argue that it is human nature to constantly strive at reducing uncertainty. The probabilities p, p' can be reduced by studying the processes involved more deeply, by having other people check the proofs, by repeating calculations, etc. In fact there is no predetermined positive lower bound for p and p'. But 4^{-100} just sits there. It is a positive constant, albeit small. It cannot be made smaller. (Of course, if you are willing to perform more strong pseudoprime tests, it can be replaced by a smaller constant.)

The probability 4^{-100} is also a little ephemeral. Say the randomly chosen bases for which we perform the strong pseudoprime tests are $b_1, \ldots, b_{100}$. And say not only does n pass these tests, but for some reason the computer *tells* us the bases $b_1, \ldots, b_{100}$ its random number generator chose. This extra information could conceivably alter the probability of error. Indeed, if we happen to recognize the set $\{b_1, \ldots, b_{100}\}$ for having an inordinately large number of strong pseudoprimes, then there may be a higher chance n is composite. In particular, say n is composite and $n \equiv 3 \bmod 4$. Then from Malm [19] it follows that the set of bases for which n is a strong pseudoprime comprise a subgroup of the multiplicative group of reduced residues mod n. Thus if we notice that the set $\{b_1, \ldots, b_{100}\}$ is generated multiplicatively modulo n by a smaller set $\{c_1, \ldots, c_k\}$, $k < 100$, then the probability of error now is 4^{-k}. It is even conceivable that $k = 1$. Of course we can negate all of this by just making sure the computer never tells us the numbers $b_1, \ldots, b_{100}$.

It is my opinion that we should not accept a probabilistic compositeness test as a proof of primality. If we did accept such "proofs", this lowering of standards could lead in some bizarre directions. There are already theorems in the literature of the sort: "There are at most finitely many counter-examples to Fermat's Last Theorem with probability 1" (see Erdös and Ulam [8]). What this statement means is that if the problem is generalized so that the particular case we are really interested in is just one of many cases, and if we put a natural probability distribution on the set of all cases, the the analogous theorem holds for so many cases that the probability is 1 that it holds for any one fixed but random case. Perhaps this analogy isn't perfect, but would those who would accept a probabilistic compositeness test as a proof of primality also put Fermat's Last Theorem into the "solved" column? Be forewarned though that Erdös and Ulam also proved that with probability 1 there are indeed counter-examples to Fermat's Last Theorem with exponent three!

A Probabilistic Primality Test

In 1876, Lucas [18] gave the following iron clad test for primality of n: if $g^x \equiv 1 \bmod n$ for $x = n - 1$, but not for x equal to any proper divisor of $n - 1$ (that is, g belongs to the exponent $n-1$ mod n), then n is prime. Indeed, it follows from Euler's theorem that if g belongs to the exponent h mod n, then $h|\varphi(n)$ where φ denotes Euler's function ($\varphi(n)$ is the number of integers in $[1, n]$ coprime to n). Thus if $h = n-1$, then $n-1 \leqslant \varphi(n) < n$, so that $\varphi(n) = n-1$. This can occur only if n is prime.

On the other hand, if n is prime, such a g as described by Lucas (called a primitive root) will exist. Indeed, the ring $\mathbb{Z}/n\mathbb{Z}$ is a field if n is prime, so the non-zero elements form a cyclic group. Pratt [26] recently used the Lucas test to show that for every prime n there is a very short proof that it is prime (a so-called "succint certificate of primality"). The trouble is *finding* the proof. To implement the Lucas test one has to find the good value of g and then prove it belongs to the exponent $n-1$. To accomplish the latter task you need to know the prime factorization of $n-1$. But as we remarked above, factoring is not an easy job.

About 10 years ago, Brillhart and Morrison [21] found a factoring algorithm for an integer m that makes use of the continued fraction expansion of $\sqrt{m}$. Although proved in practice, the continued fraction algorithm has never been proved rigorously. Recently, Dixon [6] gave a rigorous proof that if m is composite, then a simplified, but probabilistic version of the continued fraction algorithm will almost certainly provide a non-trivial factorization of m in time bounded by $O(e^{c\sqrt{\log m \log\log m}})$, where $c = 3\sqrt{2}$. Thus if one completely factors $n-1$ by repeating Dixon's algorithm (and one proves the prime factors really are prime by an iteration of the procedure), then we can almost certainly provide a rigorous proof that n is prime (if it is so) in time bounded by $O(e^{c\sqrt{\log n \log\log n}})$.

Note that once a complete factorization of $n-1$ is known one can use the following simple probabilistic algorithm for the production of a primitive root g of n. Namely, choose residues mod n at random until a primitive root is found. Since a prime $n > 3$ has

$$\varphi(n-1) > \frac{n}{6 \log\log n}$$

(see Rosser and Schoenfeld [30], p. 72) primitive roots in $[1, n]$, this random search will almost certainly be successful before $\log n$ guesses are made.

Thus we can almost certainly find a proof of primality for prime n in sub-exponential time. However, this algorithm is mainly of theoretical interest, since we cannot in practice factor numbers with much more than 40 decimal digits (Wunderlich [38]) unless we have good fortune or the number is special. On the practical level we can use other methods to prove that much larger numbers are prime, as we shall see in the next section. Even on the theoretical level we can do better as we shall see two sections hence.

The Number Crunchers

We have mentioned above that if the prime factorization of $n-1$ is known it is relatively easy to provide a proof that n is prime if it is so. The same holds if we know the prime factorization of $n+1$. The argument in this case follows from an appropriate converse of Fermat's little theorem in a certain quadratic number field. Let $M_k = 2^k - 1$, the k-th Mersenne number. Then obviously we know the prime factorization of $M_k + 1$. The famous Lucas-Lehmer primality test for Mersenne numbers can be stated (in an unusual form) as follows: for $k > 2$ we have M_k prime if and only if $7 - 4\sqrt{3}$ belongs to the exponent $(M_k + 1)/2$ modulo M_k. (The usual statement is that if $S_1 = 4$ and $S_k = S_{k-1}^2 - 2$ for $k > 1$, then for $k > 2$, M_k is prime if and only if $M_k | S_{k-1}$. This form of the test is easy to carry out since all arithmetic is in $\mathbb{Z}$, but it disguises the fact that we are actually performing a certain pseudoprime test in the field $\mathbb{Q}(\sqrt{3})$.)

By 1971 all M_k had been tested for primality for $k < 20{,}000$. The largest of the 24 primes found was M_{19937} (see Tuckerman [34]) and was the largest number for which a proof of its primality was known. This record number with 6002 decimal digits fascinated laymen. In fact, a picture of Tuckerman standing in front of his prime written in decimal was in one of the editions of the Guiness Book of World Records!

Tuckerman's world record stood for only 7 years. In October, 1978, two high school students, Laura Nickel and Curt Noll, using the computer center at California State University at Hayward found that M_{21701} is prime. This discovery not only made every major newspaper, but was accorded the ultimate honor of being reported by the television newscaster Walter Cronkite! A few months later with much less publicity, Noll (see [22]) found that M_{23209} is prime. But even this record didn't stand for long. Just 6 weeks later in April, 1979, David Slowinski [32], working on a CRAY-1 computer, checked every M_k until he found that M_{44497} is prime. This number has 13,395 decimal digits.

Most people believe there are infinitely many Mersenne primes, but this has never been proved and seems hopeless at this time. A heuristic argument (Gillies [10]) suggests there are asymptotically $c \log x$ values of $k \leqslant x$ with M_k prime. The numerical evidence appears to support this as well. Gillies' argument suggests the value $2/\log 2 \doteq 2.885$ for c. Both Lenstra and the present writer have heuristic arguments that suggest $e^{\gamma}/\log 2 \doteq 2.570$ for c (where γ is Euler's constant). This value of c appears to fit the data better as well.

In 1975, Brillhart, Lehmer, and Selfridge [3], building on earlier work of Pocklington, Lehmer, and Robinson, showed that if n is prime, a proof of its primality can often be provided even if only a partial factorization of $n-1$ or $n+1$ is known. In fact they showed how partial factorizations of both $n \pm 1$ can be used simultaneously in one primality test. Later Williams, Judd, and Holte [36], [37] extended these ideas where now, in addition, partial factorizations of $n^2 + 1$, $n^2 + n + 1$, and $n^2 - n + 1$ are thrown into the pot.

A triumph of these methods is the proof (in [36]) that the 121 digit number $2^{400} - 593$ is prime. This particular number was chosen in [27] by Rabin as a test case for his probabilistic algorithm. It is the largest prime below 2^{400} and has no special form. Rabin declared it prime after it passed over 100 randomly chosen strong pseudoprime tests. Now Williams and Holte have proved that it is prime.

A Nearly Polynomial Time Algorithm

In 1980, a breakthrough was achieved by Leonard Adleman and Robert Rumely. They invented a new primality testing algorithm that had the important feature of being able to extract extra information from a pseudoprime test other than just "pass" or "fail". This extra information could then be used to prove primality. Moreover, the pseudoprime tests were performed in various cyclotomic extensions of the rational numbers, and so the subtleties of arithmetic in these fields, such as the higher reciprocity laws, became involved. Unsure of the running time, Adleman and Rumely had a heuristic argument that it was very fast, in fact almost polynomial. Specifically they conjectured the running time was bounded by $O((\log n)^{c \log\log\log n})$ where n is the number to be tested for primality. Since $\log\log\log n$ grows so slowly, this bound is almost polynomial in the input length (which is proportional to $\log n$). Somewhat later the present writer and Andrew Odlyzko were able to prove correct their conjecture on the running time.

The Adleman-Rumely test does not suffer from the defects of the probabilistic compositeness tests or Miller's test. It will *prove* that n is prime, if it is so.

The algorithm appears to be computer practical. William Dubuque has run it on a 62 digit prime (a factor of $2^{2^8} + 1$ recently discovered by Brent and Pollard), taking about 6 hours on a moderately fast computer. The same prime was handled in just a few minutes of computer time by Hugh Williams using his algorithm. For a first attempt though, Dubuque's 6 hour running time is encouraging, especially since the algorithm should not take many times longer for a number with twice as many digits. In addition, Williams has noted that the new algorithm can be used in tandem with his own. Thus there are good causes for optimism among primality testers.

Recently Hendrik Lenstra has discovered several variations of the new algorithm. These variations bypass direct use of the higher reciprocity laws in their proofs of correctness, but instead depend on the properties of Gauss sums and Jacobi sums (the latter are also employed in the Adleman-Rumely algorithm). An exciting aspect of Lenstra's variations are that they may be easier to run on a computer.

We now give a brief description of the Adleman-Rumely algorithm. For more details see [1]. On input of the number n to be tested for primality, compute the least square-free integer $f(n)$ such that

$$\prod_{\substack{q-1 \mid f(n) \\ q \text{ prime}}} q > \sqrt{n}\,.$$

The prime factors of $f(n)$ are called *initial* primes, while the primes q with $q - 1 \mid f(n)$ are called *Euclidean* primes. Check that n is divisible by no initial or Euclidean primes. If n is composite then it has a prime factor $r \leqslant \sqrt{n}$. Since the product of the Euclidean primes exceeds $\sqrt{n}$, we can find r by the Chinese Remainder Theorem if we know each residue $r \bmod q$. If t_q denotes a primitive root for q and $\operatorname{Ind}_q(r)$ is the least positive integer with

$$r \equiv t_q^{\operatorname{Ind}_q(r)} \bmod q, \tag{2}$$

then knowing the residue $r \bmod q$ is equivalent to knowing $\operatorname{Ind}_q(r)$. However $1 \leqslant \operatorname{Ind}_q(r) \leqslant q - 1$ and $q - 1$ is a product of distinct primes which divide $f(n)$. Thus we can find $\operatorname{Ind}_q(r)$, again by the Chinese Remainder Theorem, if we know each residue $\operatorname{Ind}_q(r) \bmod p$ for each prime $p \mid q - 1$. In summary, if $\operatorname{Ind}_q(r) \bmod p$ can be found for each pair of primes p, q with $p \mid q - 1 \mid f(n)$, then we can compute r, the hypothetical prime factor of n.

Let us examine the significance of this idea for the special case $p = 2$. There are just two possible values for $\operatorname{Ind}_q(r) \bmod 2$ (where q is an odd Euclidean prime), namely 0 or 1. Moreover, from (2), the parity of $\operatorname{Ind}_q(r)$ tells us something very concrete about r. Namely, if $\operatorname{Ind}_q(r)$ is even, then r is a square mod q; if $\operatorname{Ind}_q(r)$ is odd, then r is not a square mod q. In the language of the Legendre symbol (introduced in the section on the ERH), we have

$$\left(\frac{r}{q}\right) = (-1)^{\operatorname{Ind}_q(r)} \tag{3}$$

for each odd Euclidean prime q. Thus, if we can compute (r/q), then we can compute $\operatorname{Ind}_q(r) \bmod 2$.

But, of course, we cannot compute (r/q) unless we know r, so it would seem as if we have reached a dead end. The key idea is that it is possible to *compute relationships* among the (r/q) such that if just one of them is *guessed*, then all of the others may be found in terms of it. This "explosion of information" accounts for the fast running time of the algorithm.

We first note that, by the Law of Quadratic Reciprocity, we can "flip" Legendre symbols. In particular, since $-q \equiv 1 \bmod 4$, we have

$$\left(\frac{r}{q}\right) = \left(\frac{-q}{r}\right).$$

Now we cannot compute $(-q/r)$, but we can compute the "mock residue symbol"

$$\left\langle\frac{-q}{n}\right\rangle = \begin{cases} \pm 1 \equiv (-q)^{(n-1)/2} \bmod n, & \text{if such a congruence holds} \\ \text{undefined}, & \text{otherwise.}\end{cases}$$

Thus if $\langle -q/n\rangle$ is defined, then n has passed a certain pseudoprime test to the base $-q$. If $\langle -q/n\rangle$ is undefined, then n must be composite and the algorithm may be halted. So assume all of the $\langle -q/n\rangle$ are defined for all odd Euclidean primes q. Also suppose at least one of these symbols is -1, say $\langle -q_0/n\rangle = -1$. (This assumption may turn out to be no problem in practice, but in theory, there is no reason why such a q_0 should exist and be computable. This lacuna is patched up with some difficulty in the actual algorithm.) Then it is possible to compute numbers $m_q = 0$ or 1 for each odd q such that

$$\left\langle\frac{-q_0}{n}\right\rangle^{m_q} = \left\langle\frac{-q}{n}\right\rangle. \tag{4}$$

They key idea is that this relationship is preserved if r replaces n. In fact, we have

$$\left(\frac{-q_0}{r}\right)^{m_q} = \left(\frac{-q}{r}\right). \tag{5}$$

So we still have not computed the $(-q/r)$, but we have computed the numbers m_q, and thus there are only two possibilities. If $(-q_0/r) = 1$, then each $(-q/r) = 1$. If $(-q/r) = -1$, then each $(-q/r) = (-1)^{m_q}$.

Generalizing the above argument for the case of an initial prime $p > 2$ is where the fun begins. The first goal is to generalize (3) and this is accomplished by introducing p-th power residue symbols (the Legendre symbol is the 2-nd power residue symbol). If ζ_p is a primitive p-th root of 1 and if $\mathscr{Q} + (p)$ is a prime ideal in the ring $\mathbb{Z}[\zeta_p]$, then the symbol $(\alpha/\mathscr{Q})_p$ is defined as that p-th root of 1 satisfying

$$\left(\frac{\alpha}{\mathscr{Q}}\right)_p = \zeta_p^j \equiv \alpha^{(N\mathscr{Q}-1)/p} \bmod \mathscr{Q}.$$

Here, α is an element of $\mathbb{Z}[\zeta_p]$ not in $\mathscr{Q}$ and $N\mathscr{Q}$, the norm of $\mathscr{Q}$, is the cardinality of the quotient ring $\mathbb{Z}[\zeta_p]/\mathscr{Q}$. (In addition, by letting this symbol be multiplicative in the lower argument, it can be defined for ideals $\mathscr{Q}$ that are not necessarily prime.) Suppose p, q are primes with $p|q-1|f(n)$. Then there is a certain canonical prime ideal $\mathscr{Q}_p$ in $\mathbb{Z}[\zeta_p]$ lying over q that is related to the primitive root t_q of q. For this ideal $\mathscr{Q}_q$ we have

$$\left(\frac{r}{\mathscr{Q}_q}\right)_p = \zeta_p^{\mathrm{Ind}_q(r)}$$

so we have indeed been successful in generalizing (3).

Our next problem is to try to "flip" the symbol $(r/\mathscr{Q}_q)_p$ as we did in the case $p = 2$. There is indeed a p-th power reciprocity law, but we can only use it to flip symbols where the lower argument is an element of $\mathbb{Z}[\zeta_p]$. And unfortunately $\mathbb{Z}[\zeta_p]$ need not be a principal ideal domain, so that the ideal $\mathscr{Q}_q$ need not be principally generated. We solve this problem by replacing $\mathscr{Q}_q$ with a certain element of $\mathbb{Z}[\zeta_p]$ denote $J_p(q)$. This element has several fortuitous properties. First of all it is computable (after all, we *are* describing an algorithm). Second, knowing $(r/J_p(q))_p$ allows us to compute $(r/\mathscr{Q}_q)_p$, so we lose no information in passing to $J_p(q)$. Third, the p-th power reciprocity law takes the very nice form

$$\left(\frac{r}{J_p(q)}\right)_p = \left(\frac{J_p(q)}{r}\right)_p.$$

This remarkable element $J_p(q)$ appears to be just what the doctor ordered! (For the cognoscenti, $J_p(q)$ is a certain Jacobi sum.)

The last difficulty in generalizing the $p = 2$ case is the definition of an appropriate mock residue symbol. The problem is that, even if n is a rational prime, the ideal (n) in $\mathbb{Z}[\zeta_p]$ may not be prime, so that $(J_p(q))^{(N(n)-1)/p}$ is not necessarily congruent to a $\zeta_p^j \bmod n$. We thus factor the ideal (n) in $\mathbb{Z}[\zeta_p]$ as it would factor if n were prime and use for our modulus one of these ideal factors instead of n itself. It is then possible to define a p-th power mock residue symbol and to prove an appropriate generalization of the theorem that (4) implies (5). Each time a p-th power mock residue symbol is computed, we are essentially performing a pseudoprime test on n in the ring $\mathbb{Z}[\zeta_p]$.

When all of the analysis is completed for all initial primes p, the guessing stage begins. For each p there are p possibilities and thus there are a total of $f(n)$ (which is the product of the p's) guesses. Each guess allows one to construct a potential prime factor of n. If a non-trivial prime factor is produced in this fashion, then we have proved n is composite. However, if no factors are found, then we have proved n is prime. (At first glance it would seem as if we have described a factoring algorithm. However almost certainly any n that has run the gauntlet of pseudoprime tests earlier in the algorithm is prime — the last step is not valid for any n that has earlier been found out by the algorithm as composite.)

The running time for the algorithm can be shown to be $O(f(n)^c)$. Thus it is crucial to estimate $f(n)$. This task can be accomplished by modifying a 1955 argument of Prachar that shows there are infinitely many numbers m which have an inordinate number of divisors d for which $d + 1$ is prime. In particular, it can be shown

$$f(n) \leqslant (\log n)^{c' \log\log\log n}$$

for all large n. Moreover, apart from the constant c', this estimate is best possible.

Conclusion

Is there an algorithm that can decide whether n is prime or composite and that runs in polynomial time? The Adleman-Rumely algorithm and the Lenstra variations come so close, that it would seem that almost any improvement would give the final breakthrough.

Is factoring inherently more difficult than distinguishing between primes and composites? Most people feel that this is so, but perhaps this problem too will soon yield.

In his "Disquisitiones Arithmeticae" Gauss [9], p. 396, wrote

"The problem of distinguishing prime numbers from composite numbers and of resolving the latter into their prime factors is known to be one of the most important and useful in arithmetic. It has engaged the industry and wisdom of ancient and modern geometers to such an extent that it would be superfluous to discuss the problem at length. Nevertheless we must confess that all methods that have been proposed thus far are either restricted to very special cases or are so laborious and prolix that even for numbers that do not exceed the limits of tables constructed by estimable men, i.e., for numbers that do not yield to artificial methods, they try the patience of even the practiced calculator. And these methods do not apply at all to larger numbers."

The struggle continues!

References

1. L. M. Adleman, C. Pomerance, R. S. Rumely, On distinguishing prime numbers from composite numbers (to appear). Also see the paper by L. M. Adleman of the same title in Proc. 21-st FOCS Conf. (1980)
2. N. C. Ankeny, The least quadratic non residue. Ann. Math. (2) 55 (1952), 65–72
3. J. Brillhart, D. H. Lehmer, J. L. Selfridge, New primality criteria and factorizations of $2^m \pm 1$. Math. Comp. 29 (1975), 620--647
4. R. A. DeMillo, R. J. Lipton, A. J. Perlis, Social processes and proofs of theorems and programs, Comm. ACM 22 (1979), 271–280. Also in Math. Intelligencer 3 (1980), 31–40
5. L. E. Dickson, *History of the Theory of Numbers*, Vol. 1, Press of Gibson Brothers, Washington, D.C. 1919
6. J. D. Dixon, Asymptotically fast factorization of integers. Math. Comp. 36 (1981), 255–260
7. P. Erdös, On almost primes, Amer. Math. Monthly 57 (1950), 404–407
8. P. Erdös, S. Ulam, Some probabilistic remarks on Fermat's last theorem. Rocky Mountain J. Math. 1 (1971), 613–616
9. C. F. Gauss, *Disquisitiones Arithmeticae* (A. A. Clarke, transl.). Yale University Press, New Haven, Connecticut, London 1966
10. D. B. Gillies, Three new Mersenne primes and a statistical theory. Math. Comp. 18 (1964), 93–97
11. G. B. Kolata, Mathematical proofs: The genesis of reasonable doubt. Science 192 (1976), 989–990
12. G. B. Kolata, Prior restraints on cryptogryphy considered. Science 208 (1980), 1442--1443. Also, Cryptology: A secret meeting at IDA? Science 200 (1978), 184. Also, D. Shapley and G. B. Kolata, Cryptology: Scientists puzzle over threat to open research, publication. Science 197 (1977), 1345--1349
13. I. Lakatos, *Proofs and Refutations*. Cambridge University Press, Cambridge, London, New York, Melbourne 1976
14. D. H. Lehmer, On the exact number of primes less than a given limit. Illinois J. Math. 3 (1959), 381–388
15. D. H. Lehmer, Strong Carmichael numbers. J. Austral. Math. Soc. Ser. A 21 (1976), 508–510
16. H. W. Lenstra, Jr., Miller's primality test. Inform. Process. Lett. 8 (1979), 86--88
17. H. W. Lenstra, Jr., Primality testing. Studieweek Getaltheorie en Computers, Sept. 1–5, 1980, Stichting Math. Centrum, Amsterdam
18. E. Lucas, *Théorie des Nombres*. Tome 1, Librarie Blanchard, Paris, 1961
19. D. E. G. Malm, On Monte-Carlo primality tests. Unpublished (but see [25], Theorem 4)
20. G. L. Miller, Riemann's hypothesis and tests for primality. J. Comput. System Sci. 13 (1976), 300--317
21. M. A. Morrison, J. Brillhart, A method of factoring and the factorization of F_7. Math. Comp. 29 (1975), 183--205
22. C. Noll, L. Nickel, The 25th and 26th Mersenne primes. Math. Comp. 35 (1980), 1387–1390
23. C. Pomerance, A new lower bound for the pseudoprime counting function. Illinois J. Math., to appear
24. C. Pomerance, On the distribution of pseudoprimes. Math. Comp., to appear
25. C. Pomerance, J. L. Selfridge, S. S. Wagstaff, Jr., The pseudoprimes to $25 \cdot 10^9$. Math. Comp. 35 (1980), 1003--1026
26. V. R. Pratt, Every prime has a succinct certificate. SIAM J. Comput. 4 (1975), 214--220
27. M. O. Rabin, Probabilistic algorithms. In: J. F. Traub, ed., *Algorithms and Complexity*. Academic Press, New York, San Francisco, London, 1976, 21–39
28. M. O. Rabin, Probabilistic algorithm for primality testing. J. Number Theory 12 (1980), 128–138
29. R. Rivest, A. Shamir, L. Adleman, A method for obtaining digital signatures and public-key cryptosystems. MIT Lab for Comp. Sci., Technical Memo LCS/TM82, 1977. Also Comm. ACM 21 (1978), 120--128
30. J. B. Rosser, L. Schoenfeld, Approximate formulas for some functions of prime numbers. Illinois J. Math. 6 (1962), 64--94
31. G. J. Simmons, Cryptology: The mathematics of secure communication. Math. Intelligencer 1 (1979), 233–246
32. D. Slowinski, Searching for the 27th Mersenne prime. J. Recreational Math. 11 (1978–79), 258–261
33. R. Solovay, V. Strassen, A fast Monte-Carlo test for primality. SIAM J. Comput. 6 (1977), 84–85; erratum. 7 (1978), 118
34. B. Tuckerman, The 24th Mersenne prime. Proc. Nat. Acad. Sci. USA 68 (1971), 2319–2320
35. H. C. Williams, Primality testing on a computer. Ars Combinatoria 5 (1978), 127–185
36. H. C. Williams, R. Holte, Some observations on primality testing, Math. Comp. 32 (1978), 905--917
37. H. C. Williams, J. S. Judd, Some algorithms for prime test-