A Tale of Two Sieves

Carl Pomerance

(This paper is dedicated to the memory of my friend and teacher, Paul Erdős)

It is the best of times for the game of factoring large numbers into their prime factors. In 1970 it was barely possible to factor "hard" 20-digit numbers. In 1980, in the heyday of the Brillhart-Morrison *continued fraction factoring algorithm*, factoring of 50-digit numbers was becoming commonplace. In 1990 my own *quadratic sieve factoring algorithm* had doubled the length of the numbers that could be factored, the record having 116 digits.

By 1994 the quadratic sieve had factored the famous 129-digit RSA challenge number that had been estimated in Martin Gardner's 1976 *Scientific American* column to be safe for 40 quadrillion years (though other estimates around then were more modest). But the quadratic sieve is no longer the champion. It was replaced by Pollard's *number field sieve* in the spring of 1996, when that method successfully split a 130-digit RSA challenge number in about 15% of the time the quadratic sieve would have taken.

In this article we shall briefly meet these factorization algorithms—these two sieves—and some of the many people who helped to develop them.

In the middle part of this century, computational issues seemed to be out of fashion. In most books the problem of factoring big numbers was largely ignored, since it was considered trivial. After all, it was doable *in principle*, so what else was there to discuss? A few researchers ignored the fashions of the time and continued to try to find fast ways to factor. To these few it was a basic and fundamental problem, one that should not be shunted to the side.

But times change. In the last few decades we have seen the advent of accessible and fast computing power, and we have seen the rise of cryptographic systems that base their security on our supposed inability to factor quickly (and on other number theoretic problems). Today there are many people interested in factoring, recognizing it not only as a benchmark for the security of cryptographic systems, but for computing itself. In 1984 the Association for Computing Machinery presented a plaque to the Institute for Electrical and Electronics Engineers (IEEE) on the occasion of the IEEE centennial. It was inscribed with the prime factorization of the number $2^{251} - 1$ that was completed that year with the quadratic sieve. The president of the ACM made the following remarks:

> About 300 years ago the French mathematician Mersenne speculated that $2^{251} - 1$ was a composite, that is, a factorable number. About 100 years ago it was proved to be factorable, but even 20 years ago the computational load to factor the number was considered insurmountable. Indeed, using conventional machines and traditional search algorithms, the search time

Carl Pomerance is research professor of mathematics at the University of Georgia, Athens, GA. His e-mail address is `carl@ada.math.uga.edu`.

was estimated to be about 10^{20} years. The number was factored in February of this year at Sandia on a Cray computer in 32 hours, a world record. We've come a long way in computing, and to commemorate IEEE's contribution to computing we have inscribed the five factors of the Mersenne composite on a plaque. Happy Birthday, IEEE.

Factoring big numbers is a strange kind of mathematics that closely resembles the experimental sciences, where nature has the last and definitive word. If some method to factor n runs for awhile and ends with the statement *"d is a factor of n"*, then this assertion may be easily checked; that is, the integers have the last and definitive word. One can thus get by quite nicely without proving a theorem that a method works in general. But, as with the experimental sciences, both rigorous and heuristic analyses can be valuable in understanding the subject and moving it forward. And, as with the experimental sciences, there is sometimes a tension between pure and applied practitioners. It is held by some that the theoretical study of factoring is a freeloader at the table (or as Hendrik Lenstra once colorfully put it, paraphrasing Siegel, "a pig in the rose garden"), enjoying undeserved attention by vapidly giving various algorithms labels, be they "polynomial", "exponential", "random", etc., and offering little or nothing in return to those hard workers who would seriously compute. There is an element of truth to this view. But as we shall see, theory played a significant role in the development of the title's two sieves.

A Contest Problem

But let us begin at the beginning, at least *my* beginning. When I give talks on factoring, I often repeat an incident that happened to me long ago in high school. I was involved in a math contest, and one of the problems was to factor the number 8051. A time limit of five minutes was given. It is not that we were not allowed to use pocket calculators; they did not exist in 1960, around when this event occurred! Well, I was fairly good at arithmetic, and I was sure I could trial divide up to the square root of 8051 (about 90) in the time allowed. But on any test, especially a contest, many students try to get into the mind of the person who made it up. Surely they would not give a problem where the only reasonable approach was to try possible divisors frantically until one was found. There must be a clever alternate route to the answer. So I spent a couple of minutes looking for the clever way, but grew worried that I was wasting too much time. I then belatedly started trial division, but

I *had* wasted too much time, and I missed the problem.

So can you find the clever way? If you wish to think about this for a moment, delay reading the next paragraph.

Fermat and Kraitchik

The trick is to write 8051 as $8100 - 49$, which is $90^2 - 7^2$, so we may use algebra, namely, factoring a difference of squares, to factor 8051. It is 83×97.

Does this always work? In fact, every odd composite can be factored as a difference of squares: just use the identity $ab = \left(\frac{1}{2}(a+b)\right)^2 - \left(\frac{1}{2}(a-b)\right)^2$. Trying to find a pair of squares which work is, in fact, a factorization method of Fermat. Just like trial division, which has some very easy cases (such as when there is a small prime factor), so too does the difference-of-squares method have easy cases. For example, if $n = ab$ where a and b are very close to $\sqrt{n}$, as in the case of $n = 8051$, it is easy to find the two squares. But in its worst cases, the difference-of-squares method can be far worse than trial division. It is worse in another way too. With trial division, most numbers fall into the easy case; namely, most numbers have a small factor. But with the difference-of-squares method, only a small fraction of numbers have a divisor near their square root, so the method works well on only a small fraction of possible inputs. (Though trial division allows one to *begin* a factorization for most inputs, finishing with a complete factorization is usually far more difficult. Most numbers resist this, even when a combination of trial division and difference-of-squares is used.)

In the 1920s Maurice Kraitchik came up with an interesting enhancement of Fermat's difference-of-squares technique, and it is this enhancement that is at the basis of most modern factoring algorithms. (The idea had roots in the work of Gauss and Seelhoff, but it was Kraitchik who brought it out of the shadows, introducing it to a new generation in a new century. For more on the early history of factoring, see [23].) Instead of trying to find integers u and v with $u^2 - v^2$ equal to n, Kraitchik reasoned that it might suffice to find u and v with $u^2 - v^2$ equal to a *multiple* of n, that is, $u^2 \equiv v^2 \bmod n$. Such a congruence can have uninteresting solutions, those where $u \equiv \pm v \bmod n$, and interesting solutions, where $u \not\equiv \pm v \bmod n$. In fact, if n is odd and divisible by at least two different primes, then at least half of the solutions to $u^2 \equiv v^2 \bmod n$, with uv coprime to n, are of the interesting variety. And for an interesting solution u, v, the greatest common factor of $u - v$ and n, denoted $(u - v, n)$, must be a nontrivial factor of n. Indeed, n divides $u^2 - v^2 =$

$(u - v)(u + v)$ but divides neither factor. So n must be somehow split between $u - v$ and $u + v$.

As an aside, it should be remarked that finding the greatest common divisor (a, b) of two given numbers a and b is a very easy task. If $0 < a \leq b$ and if a divides b, then $(a, b) = a$. If a does not divide b, with b leaving a remainder r when divided by a, then $(a, b) = (a, r)$. This neat idea of replacing a larger problem with a smaller one is over two thousand years old and is due to Euclid. It is very fast: it takes about as much time for a computer to find the greatest common divisor of a and b as it would take to multiply them together.

Let us see how Kraitchik might have factored $n = 2041$. The first square above n is $46^2 = 2116$. Consider the sequence of numbers $Q(x) = x^2 - n$ for $x = 46, 47, \ldots$. We get

$$75, \ 168, \ 263, \ 360, \ 459, \ 560, \ldots.$$

So far no squares have appeared, so Fermat might still be searching. But Kraitchik has another option: namely, he tries to find several numbers x with the product of the corresponding numbers $Q(x)$ equal to a square. For if $Q(x_1) \cdots Q(x_k) = v^2$ and $x_1 \cdots x_k = u$, then

$$u^2 = x_1^2 \cdots x_k^2 \equiv (x_1^2 - n) \cdots (x_k^2 - n)$$
$$= Q(x_1) \cdots Q(x_k) = v^2 \bmod n;$$

that is, we have found a solution to $u^2 \equiv v^2 \bmod n$. But how to find the set $x_1, \ldots, x_k$? Kraitchik notices that some of the numbers $Q(x)$ factor *very* easily:

$$75 = 3 \times 5^2, \quad 168 = 2^3 \times 3 \times 7,$$
$$360 = 2^3 \times 3^2 \times 5, \quad 560 = 2^4 \times 5 \times 7.$$

From these factorizations he can tell that the product of these four numbers is $2^{10} \times 3^4 \times 5^4 \times 7^2$, a square! Thus he has $u^2 \equiv v^2 \bmod n$, where

$$u = 46 \cdot 47 \cdot 49 \cdot 51 \equiv 311 \bmod 2041,$$
$$v = 2^5 \cdot 3^2 \cdot 5^2 \cdot 7 \equiv 1416 \bmod 2041.$$

He is now nearly done, since $311 \not\equiv \pm 1416 \bmod 2041$. Using Euclid's algorithm to compute the greatest common factor $(1416 - 311, 2041)$, he finds that this is 13, and so $2041 = 13 \times 157$.

Continued Fractions

The essence of Kraitchik's method is to "play" with the sequence $x^2 - n$ as x runs through integers near $\sqrt{n}$ to find a subsequence with product a square. If the square root of this square is v and the product of the corresponding x values is u, then $u^2 \equiv v^2 \bmod n$, and there is now a hope that this congruence is "interesting", namely, that $u \not\equiv \pm v \bmod n$. In 1931 D. H.

Lehmer and R. E. Powers suggested replacing Kraitchik's function $Q(x) = x^2 - n$ with another that is derived from the continued-fraction expansion of $\sqrt{n}$.

If a_i / b_i is the i-th continued fraction convergent to $\sqrt{n}$, let $Q_i = a_i^2 - b_i^2 n$. Then $Q_i \equiv a_i^2 \bmod n$. Thus, instead of playing with the numbers $Q(x)$, we may play with the numbers Q_i, since in both cases they are congruent modulo n to known squares. Although continued fractions can be ornery beasts as far as computation goes, the case for quadratic irrationals is quite pleasant. In fact, there is a simple iterative procedure (see [16]) going back to Gauss and perhaps earlier for computing what is needed here, namely, the sequence of integers Q_i and the residues $a_i \bmod n$.

But why mess up a perfectly simple quadratic polynomial with something as exotic as continued fractions? It is because of the inequality $|Q_i| < 2\sqrt{n}$. The numbers Q_i are smaller in absolute value than the numbers $Q(x)$. (As x moves away from $\sqrt{n}$, the numbers $Q(x)$ grow approximately linearly, with a slope of $2\sqrt{n}$.) If one wishes to "play" with numbers to find some of them with product a square, it is presumably easier to do this with smaller numbers than with larger numbers. So the continued fraction of Lehmer and Powers has an apparent advantage over the quadratic polynomial of Kraitchik.

How to "Play" with Numbers

It is certainly odd to have an instruction in an algorithm asking you to play with some numbers to find a subset with product a square. I am reminded of the cartoon with two white-coated scientists standing at a blackboard filled with arcane notation, and one points to a particularly delicate juncture and says to the other that at this point a miracle occurs. Is it a miracle that we were able to find the numbers 75, 168, 360, and 560 in Kraitchik's sequence with product a square? Why should we expect to find such a subsequence, and, if it exists, how can we find it efficiently?

A systematic strategy for finding a subsequence of a given sequence with product a square was found by John Brillhart and Michael Morrison, and, surprisingly, it is at heart only linear algebra (see [16]). Every positive integer m has an *exponent vector* $v(m)$ that is based on the prime factorization of m. Let p_i denote the i-th prime, and say $m = \prod p_i^{v_i}$. (The product is over all primes, but only finitely many of the exponents v_i are nonzero.) Then $v(m)$ is the vector $(v_1, v_2, \ldots)$. For example, leaving off the infinite string of zeros after the fourth place, we have

$$\nu(75) = (0, 1, 2, 0),$$
$$\nu(168) = (3, 1, 0, 1),$$
$$\nu(360) = (3, 2, 1, 0),$$
$$\nu(560) = (4, 0, 1, 1).$$

For our purposes the exponent vectors give *too much* information. We are interested only in squares, and since a positive integer m is a square if and only if every entry of $\nu(m)$ is even, we should be reducing exponents modulo 2. Since ν takes products to sums, we are looking for numbers such that the sum of their exponent vectors is the zero vector mod 2. The mod 2 reductions of the above exponent vectors are

$$\nu(75) \equiv (0, 1, 0, 0) \bmod 2,$$
$$\nu(168) \equiv (1, 1, 0, 1) \bmod 2,$$
$$\nu(360) \equiv (1, 0, 1, 0) \bmod 2,$$
$$\nu(560) \equiv (0, 0, 1, 1) \bmod 2.$$

Note that their sum is the zero vector! Thus the product of 75, 168, 360, and 560 is a square.

To systematize this procedure, Brillhart and Morrison suggest that we choose some number B and look only at those numbers in the sequence that completely factor over the first B primes. So in the case above, we have $B = 4$. As soon as $B + 1$ such numbers have been assembled, we have $B + 1$ vectors in the B-dimensional vector space $\mathbb{F}_2^B$. By linear algebra they must be linearly dependent. But what is a linear dependence relation over the field $\mathbb{F}_2$? Since the only scalars are 0 and 1, a linear dependence relation is simply a subset sum equaling the 0-vector. And we have many algorithms from linear algebra that can help us find this dependency.

Note that we were a little lucky here, since we were able to find the dependency with just four vectors rather than the five vectors needed in the worst case.

Brillhart and Morrison call the primes $p_1, p_2, \ldots, p_B$ the "factor base". (To be more precise, they discard those primes p_j for which n is not congruent to a square, since such primes will never divide a number Q_i in the continued-fraction method nor a number $Q(x)$ in Kraitchik's method.) How is B to be chosen? If it is chosen small, then we do not need to assemble too many numbers before we can stop. But if it is chosen too small, the likelihood of finding a number in the sequence that completely factors over the first B primes will be so minuscule that it will be difficult to find even one number. Thus somewhere there is a happy balance, and with factoring 2041 via Kraitchik's method, the happy balance turned out to be $B = 4$.

Some of the auxiliary numbers may be negative. How do we handle their exponent vectors? Clearly we cannot ignore the sign, since squares are not negative. However, we can put an extra coordinate in each exponent vector, one that is 0 for positive numbers and 1 for negative numbers. (It is as if we are including the "prime" -1 in the factor base.) So allowing the auxiliary numbers to be negative just increases the dimension of the problem by 1.

For example, let us consider again the number 2041 and try to factor it via Kraitchik's polynomial, but now allowing negative values. So with $Q(x) = x^2 - 2041$ and the factor base 2, 3, and 5, we have

$$Q(43) = -192 = -2^6 \cdot 3 \leftrightarrow (1, 0, 1, 0)$$
$$Q(44) = -105$$
$$Q(45) = -16 = -2^4 \leftrightarrow (1, 0, 0, 0)$$
$$Q(46) = 75 = 3 \cdot 5^2 \leftrightarrow (0, 0, 1, 0),$$

where the first coordinates correspond to the exponent on -1. So, using the smaller factor base of 2, 3, and 5 but allowing also negatives, we are especially lucky, since the three vectors assembled so far are dependent. This leads to the congruence $(43 \cdot 45 \cdot 46)^2 \equiv (-192)(-16)(75) \bmod 2041$, or $1247^2 \equiv 480^2 \bmod 2041$. This again gives us the divisor 13 of 2041, since $(1247 - 480, 2041) = 13$.

Does the final greatest common divisor step always lead to a nontrivial factorization? No it does not. The reader is invited to try still another assemblage of a square in connection with 2041. This one involves $Q(x)$ for $x = 41$, 45, and 49 and gives rise to the congruence $601^2 \equiv 1440^2 \bmod 2041$. In our earlier terminology, this congruence is uninteresting, since $601 \equiv -1440 \bmod 2041$. And sure enough, the greatest common divisor $(601 - 1440, 2041)$ is the quite uninteresting divisor 1.

Smooth Numbers and the Stirrings of Complexity Theory

With the advent of the RSA public key cryptosystem in the late 1970s, it became particularly important to try to predict just how hard factoring is. Not only should we know the state of the art at present, we would like to predict just what it would take to factor larger numbers beyond what is feasible now. In particular, it seems empirically that dollar for dollar computers double their speed and capacity every one and a half to two years. Assuming this and no new factoring algorithms, what will be the state of the art in ten years?

It is to this type of question that complexity theory is well suited. So how then might one analyze factoring via Kraitchik's polynomial or the Lehmer-Powers continued fraction? Richard Schroeppel, in unpublished correspondence in the late 1970s, suggested a way. Essentially, he begins by thinking of the numbers Q_i in the continued-fraction method or the numbers $Q(x)$ in

Kraitchik's method as "random". If you are presented with a stream of truly random numbers below a particular bound X, how long should you expect to wait before you find some subset with product a square?

Call a number Y-smooth if it has no prime factor exceeding Y. (Thus a number which completely factors over the primes up to p_B is p_B-smooth.) What is the probability that a random positive integer up to X is Y-smooth? It is $\psi(X, Y)/[X] \approx \psi(X, Y)/X$, where $\psi(X, Y)$ is the number of Y-smooth numbers in the interval $[1, X]$. Thus the expected number of random numbers that must be examined to find just one that is Y-smooth is the reciprocal of this probability, namely, $X/\psi(X, Y)$. But we must find about $\pi(Y)$ such Y-smooth numbers, where $\pi(Y)$ denotes the number of primes up to Y. So the expected number of random numbers that must be examined is about $\pi(Y)X/\psi(X, Y)$. And how much work does it take to examine a number to see if it is Y-smooth? If one uses trial division for this task, it takes about $\pi(Y)$ steps. So the expected number of steps is $\pi(Y)^2 X/\psi(X, Y)$.

It is now a job for analytic number theory to choose Y as a function of X so as to minimize the expression $\pi(Y)^2 X/\psi(X, Y)$. In fact, in the late 1970s the tools did not quite exist to make this estimation accurately.

This was remedied in a paper in 1983 (see [4]), though preprints of this paper were around for several years before then. So what is the minimum? It occurs when Y is about $\exp(\frac{1}{2}\sqrt{\log X \log \log X})$ and the minimum value is about $\exp(2\sqrt{\log X \log \log X})$. But what are "$X$" and "$Y$" anyway?[1] The number X is an estimate for the typical auxiliary number the algorithm produces. In the continued-fraction method, X can be taken as $2\sqrt{n}$. With Kraitchik's polynomial, X is a little larger: it is $n^{1/2+\varepsilon}$. And the number Y is an estimate for p_B, the largest prime in the factor base.

Thus factoring n, either via the Lehmer-Powers continued fraction or via the Kraitchik polynomial, should take about $\exp(\sqrt{2 \log n \log \log n})$ steps. This is not a theorem; it is a conjecture. The conjecture is supported by the above heuristic argument which assumes that the auxiliary numbers generated by the continued fraction of $\sqrt{n}$ or by Kraitchik's quadratic polynomial are "random" as far as the property of being Y-smooth goes. This has not been proved. In addition, getting many auxiliary numbers that are Y-smooth may not be sufficient for factoring n, since each time we use linear algebra over $\mathbb{F}_2$ to

---

[1]*Actually, this is a question that has perplexed many a student in elementary algebra, not to mention many a philosopher of mathematics.*

assemble the congruent squares we may be very unlucky and only come up with uninteresting solutions which do not help in the factorization. Again assuming randomness, we do not expect inordinately long strings of bad luck, and this heuristic again supports the conjecture.

As mentioned, this complexity argument was first made by Richard Schroeppel in unpublished work in the late 1970s. (He assumed the result mentioned above from [4], even though at that time it was not a theorem or even really a conjecture.) Armed with the tools to study complexity, he used them during this time to come up with a new method that came to be known as the *linear sieve*. It was the forerunner of the quadratic sieve and also its inspiration.

Using Complexity to Come Up With a Better Algorithm: The Quadratic Sieve

The above complexity sketch shows a place where we might gain some improvement. It is the time we are taking to recognize auxiliary numbers that factor completely with the primes up to $Y = p_B$, that is, the Y-smooth numbers. In the argument we assumed this is about $\pi(Y)$ steps, where $\pi(Y)$ is the number of primes up to Y. The probability that a number is Y-smooth is, according to the notation above, $\psi(X, Y)/[X]$. As you might expect and as is easily checked in practice, when Y is a reasonable size and X is very large, this probability is very, very small. So one after the other, the auxiliary numbers pop up, and we have to invest all this time in each one, only to find out almost always that the number is not Y-smooth and is thus a number that we will discard.

It occurred to me early in 1981 that one might use something akin to the sieve of Eratosthenes to quickly recognize the smooth values of Kraitchik's quadratic polynomial $Q(x) = x^2 - n$. The sieve of Eratosthenes is the well-known device for finding all the primes in an initial interval of the natural numbers. One circles the first prime 2 and then crosses off every second number, namely, 4, 6, 8, etc. The next unmarked number is 3. It is circled, and we then cross off every third number. And so on. After reaching the square root of the upper limit of the sieve, one can stop the procedure and circle every remaining unmarked number. The circled numbers are the primes; the crossed-off numbers the composites.

It should be noted that the sieve of Eratosthenes does more than find primes. Some crossed-off numbers are crossed off many times. For example, 30 is crossed off three times, as is 42, since these numbers have three prime factors. Thus we can quickly scan the array looking for numbers that are crossed off a lot and so quickly find the numbers which have many

prime factors. And clearly there is a correlation between having many prime factors and having all small prime factors.

But we can do better than have a correlation. By dividing by the prime, instead of crossing off, numbers like 30 and 42 get transformed to the number 1 at the end of the sieve, since they are completely factored by the primes used in the sieve. So instead of sieving with the primes up to the square root of the upper bound of the sieve, say we only sieve with the primes up to Y. And instead of crossing a number off, we divide it by the prime. At the end of the sieve any number that has been changed to the number 1 is Y-smooth. But not every Y-smooth is caught in this sieve. For example, 60 gets divided by its prime factors and is changed to the number 2. The problem is higher powers of the primes up to Y. We can rectify this by also sieving by these higher powers and dividing hits by the underlying prime. Then the residual 1's at the end correspond exactly to the Y-smooth numbers in the interval.

The time for doing this is unbelievably fast compared with trial dividing each candidate number to see if it is Y-smooth. If the length of the interval is N, the number of steps is only about $N \log \log Y$, or about $\log \log Y$ steps on average per candidate.

So we can quickly recognize Y-smooth numbers in an initial interval. But can we use this idea to recognize Y-smooth values of the quadratic polynomial $Q(x) = x^2 - n$? What it takes for a sieve to work is that for each modulus m in the sieve, the multiples of the number m appear in regular places in the array. So take a prime p, for example, and ask, For which values of x do we have $Q(x)$ divisible by p? This is not a difficult problem. If n (the number being factored) is a nonzero square modulo p, then there are two residue classes a and b mod p such that $Q(x) \equiv 0 \bmod p$ if and only if $x \equiv a$ or b mod p. If n is not a square modulo p, then $Q(x)$ is never divisible by p and no further computations with p need be done.

So essentially the same idea can be used, and we can recognize the Y-smooth values of $Q(x)$ in about $\log \log Y$ steps per candidate value.

What does the complexity argument give us? The time to factor n is now about $\exp(\sqrt{\log n \log \log n})$; namely, the factor $\sqrt{2}$ in the exponent is missing. Is this a big deal? You bet. This lower complexity and other friendly features of the method allowed a twofold increase in the length of the numbers that could be factored (compared with the continued-fraction method discussed above). And so was born the quadratic sieve method as a complexity argument and with no numerical experiments.

Implementations and Enhancements

In fact, I was very lucky that the quadratic sieve turned out to be a competitive algorithm. More often than not, when one invents algorithms solely via complexity arguments and thought experiments, the result is likely to be too awkward to be a competitive method. In addition, even if the basic idea is sound, there well could be important enhancements waiting to be discovered by the people who actually try the thing out. This in fact happened with the quadratic sieve.

The first person to try out the quadratic sieve method on a big number was Joseph Gerver (see [9]). Using the task as an opportunity to learn programming, he successfully factored a 47-digit number from the Cunningham project. This project, begun early in this century by Lt.-Col. Allan J. Cunningham and H. J. Woodall, consists of factoring into primes the numbers $b^n \pm 1$ for b up to 12 (and not a power) and n up to high numbers (see [3]). Gerver's number was a factor of $3^{225} - 1$.

Actually I had a hard time getting people to try out the quadratic sieve. Many Cunningham project factorers seemed satisfied with the continued-fraction method, and they thought that the larger values of Kraitchik's polynomial $Q(x)$, compared with the numbers Q_i in the continued-fraction method, was too great a handicap for the fledgling quadratic sieve method. But at a conference in Winnipeg in the fall of 1982, I convinced Gus Simmons and Tony Warnock of Sandia Laboratories to give it a try on their Cray computer.

Jim Davis and Diane Holdridge were assigned the task of coding up the quadratic sieve on the Sandia Cray. Not only did they succeed, but they quickly began setting records. And Davis found an important enhancement that mitigated the handicap mentioned above. He found a way of switching to other quadratic polynomials after values of the first one, $Q(x) = x^2 - n$, grew uncomfortably large. Though this idea did not substantially change the complexity estimate, it made the method much more practical. Their success not only made the cover of the *Mathematical Intelligencer* (the volume 6, number 3 cover in 1984 had on it a Cray computer and the factorization of the number consisting of 71 ones), but there was even a short article in *Time* magazine, complete with a photo of Simmons.

It was ironic that shortly before the Sandia team began this project, another Sandia team had designed and manufactured an RSA chip for public key cryptography, whose security was based on our inability to factor numbers of about 100 digits. Clearly this was not safe enough, and the chip had to be scrapped.

Around this time Peter Montgomery independently came up with another, slightly better way of changing polynomials, and we now use his method rather than that of Davis.

One great advantage of the quadratic sieve method over the continued-fraction method is that with the quadratic sieve it is especially easy to distribute the task of factoring to many computers. For example, using multiple polynomials, each computer can be given its own set of quadratic polynomials to sieve. At first, the greatest successes of the quadratic sieve came from supercomputers, such as the Cray XMP at Sandia Laboratories. But with the proliferation of low-cost workstations and PCs and the natural way that the quadratic sieve can be distributed, the records passed on to those who organized distributed attacks on target numbers.

Robert Silverman was the first to factor a number using many computers. Later, Red Alford and I used over 100 very primitive, non-networked PCs to factor a couple of 100-digit numbers (see [2]). But we did not set a record, because while we were tooling up, Arjen Lenstra and Mark Manasse [12] took the ultimate step in distributing the problem. They put the quadratic sieve on the Internet, soliciting computer time from people all over the world. It was via such a shared effort that the 129-digit RSA challenge number was eventually factored in 1994. This project, led by Derek Atkins, Michael Graff, Paul Leyland, and Lenstra, took about eight months of real time and involved over 10^{17} elementary steps.

The quadratic sieve is ultimately a very simple algorithm, and this is one of its strengths. Due to its simplicity one might think that it could be possible to design a special-purpose computer solely dedicated to factoring big numbers. Jeff Smith and Sam Wagstaff at the University of Georgia had built a special-purpose processor to implement the continued-fraction method. Dubbed the "Georgia Cracker", it had some limited success but was overshadowed by quadratic sieve factorizations on conventional computers. Smith, Randy Tuler, and I (see [21]) thought we might build a special-purpose quadratic sieve processor. "Quasimodo", for Quadratic Sieve Motor, was built but never functioned properly. The point later became moot due to the exponential spread of low-cost, high-quality computers.

The Dawn of the Number Field Sieve

Taking his inspiration from a discrete logarithm algorithm of Don Coppersmith, Andrew Odlyzko, and Richard Schroeppel [6] that used quadratic number fields, John Pollard in 1988 circulated a letter to several people outlining an idea of his for factoring certain big numbers via algebraic
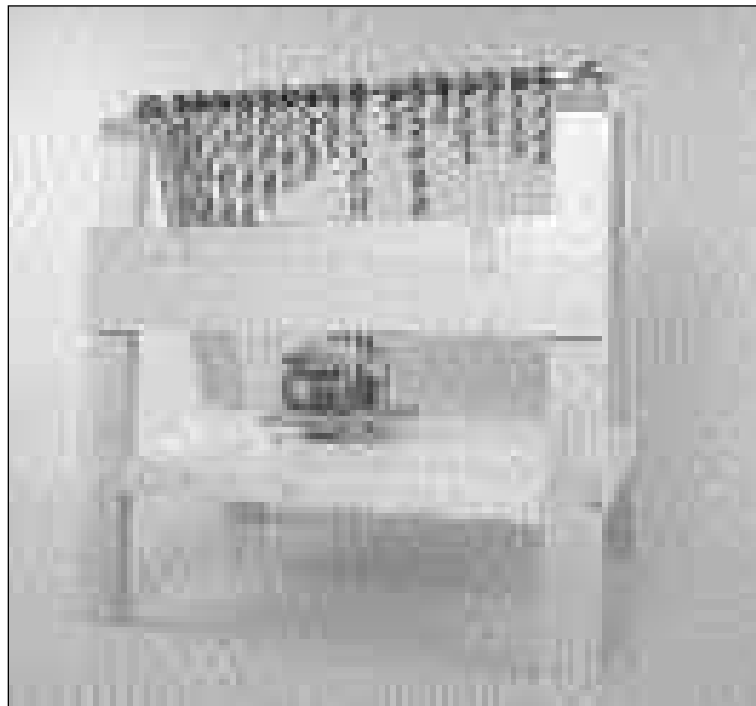
Modern model of the Lehmer Bicycle Chain Sieve constructed by Robert Canepa and currently in storage at the Computer Museum, 300 Congress Street, Boston, MA 02210.

number fields. His original idea was not for any large composite, but for certain "pretty" composites that had the property that they were close to powers and had certain other nice properties as well. He illustrated the idea with a factorization of the number $2^{2^7} + 1$, the seventh Fermat number. It is interesting that this number was the first major success of the continued-fraction factoring method, almost twenty years earlier.

I must admit that at first I was not too keen on Pollard's method, since it seemed to be applicable to only a few numbers. However, some people were taking it seriously, one being Hendrik Lenstra. He improved some details in the algorithm and, along with his brother Arjen and Mark Manasse, set about using the method to factor several large numbers from the Cunningham project. After a few successes (most notably a 138-digit number) and after Brian LaMacchia and Andrew Odlyzko had made some inroads in dealing with the large, sparse matrices that come up in the method, the Lenstras and Manasse set their eyes on a real prize, $2^{2^9} + 1$, the ninth Fermat number.[2] Clearly it was beyond the range of the quadratic sieve. Hendrik Lenstra's own elliptic curve method, which he discovered early

---

[2] *This number had already been suggested in Pollard's original note as a worthy goal. It was known to be composite—in fact, we already knew a 7-digit prime factor—but the remaining 148-digit cofactor was still composite, with no factor known.*

in 1985 and which is especially good at splitting numbers which have a relatively small prime factor (say, "only" 30 or so digits) had so far not been of help in factoring it. The Lenstras and Manasse succeeded in getting the prime factorization of $2^{2^9}+1$ in the spring of 1990. This sensational achievement announced to the world that Pollard's number field sieve had arrived.

But what of general numbers? In the summer of 1989 I was to give a talk at the meeting of the Canadian Number Theory Association in Vancouver. It was to be a survey talk on factoring, and I figured it would be a good idea to mention Pollard's new method. On the plane on the way to the meeting I did a complexity analysis of the method as to how it would work for general numbers, assuming myriad technical difficulties did not exist and that it was possible to run it for general numbers. I was astounded. The complexity for this algorithm-that-did-not-yet-exist was of the shape $\exp(c(\log n)^{1/3}$ $(\log \log n)^{2/3})$. The key difference over the complexity of the quadratic sieve was that the most important quantity in the exponent, the power of $\log n$, had its exponent reduced from $1/2$ to $1/3$. If reducing the *constant* in the exponent had such a profound impact in passing from the continued-fraction method to the quadratic sieve, think what reducing the *exponent* in the exponent might accomplish. Clearly this method deserved some serious thought!

I do not wish to give the impression that with this complexity analysis I had single-handedly found a way to apply the number field sieve to general composites. Far from it. I merely had a shrouded glimpse of exciting possibilities for the future. That these possibilities were ever realized was mostly due to Joe Buhler, Hendrik Lenstra, and others. In addition, some months earlier Lenstra had done a complexity analysis for Pollard's method applied to special numbers, and he too arrived at the expression $\exp(c(\log n)^{1/3}$ $(\log \log n)^{2/3})$. My own analysis was based on some optimistic algebraic assumptions and on arguments about what might be expected to hold, via averaging arguments, for a general number.

The starting point of Pollard's method to factor n is to come up with a monic polynomial $f(x)$ over the integers that is irreducible and an integer m such that $f(m) \equiv 0 \bmod n$. The polynomial should have "moderate" degree d, meaning that if n has between 100 and 200 digits, then d should be 5 or 6. For a number such as the ninth Fermat number, $n = 2^{2^9}+1$, it is easy to come up with such a polynomial. Note that $8n = 2^{515}+8$. So let $f(x) = x^5+8$, and let $m = 2^{103}$.

Of what possible use could such a polynomial be? Let α be a complex root of $f(x)$, and consider the ring $\mathbb{Z}[\alpha]$ consisting of all polynomial expressions in α with integer coefficients. Since $f(\alpha) = 0$ and $f(m) \equiv 0 \bmod n$, by substituting the residue $m \bmod n$ for each occurrence of α we have a natural map ϕ from $\mathbb{Z}[\alpha]$ to $\mathbb{Z}/(n\mathbb{Z})$. Our conditions on f, α, and m ensure that ϕ is well defined. And not only this, ϕ is a ring homomorphism.

Suppose now that S is a finite set of coprime integer pairs $\langle a, b \rangle$ with two properties. The first is that the product of the algebraic integers $a - \alpha b$ for all pairs $\langle a, b \rangle$ in S is a square in $\mathbb{Z}[\alpha]$, say, γ^2. The second property for S is that the product of all the numbers $a - mb$ for pairs $\langle a, b \rangle$ in S is a square in $\mathbb{Z}$, say, v^2. Since γ may be written as a polynomial expression in α, we may replace each occurrence of α with the integer m, coming up with an integer u with $\phi(\gamma) \equiv u \bmod n$. Then

$$u^2 \equiv \phi(\gamma)^2 = \phi(\gamma^2) = \phi\left(\prod_{\langle a,b \rangle \in \mathbb{S}} (a - \alpha b)\right)$$

$$= \prod_{\langle a,b \rangle \in \mathbb{S}} \phi(a - \alpha b)$$

$$\equiv \prod_{\langle a,b \rangle \in \mathbb{S}} (a - mb) = v^2 \bmod n.$$

And we know what to do with u and v. Just as Kraitchik showed us seventy years ago, we hope that we have an interesting congruence, that is, $u \not\equiv \pm v \bmod n$, and if so, we take the greatest common divisor $(u - v, n)$ to get a nontrivial factor of n.

Where is the set S of pairs $\langle a, b \rangle$ supposed to come from? For at least the second property S is supposed to have, namely, that the product of the numbers $a - mb$ is a square, it is clear we might again use exponent vectors and a sieve. Here there are two variables a and b instead of just the one variable in $Q(x)$ in the quadratic sieve. So we view this as a parametrized family of linear polynomials. We can fix b and let a run over an interval, then change to the next b and repeat.

But S is to have a second property too: for the same pairs $\langle a, b \rangle$, the product of $a - \alpha b$ is a square in $\mathbb{Z}[\alpha]$. It was Pollard's thought that if we were in the nice situation that $\mathbb{Z}[\alpha]$ is the full ring of algebraic integers in $\mathbb{Q}(\alpha)$, if the ring is a unique factorization domain, and if we know a basis for the units, then we could equally well create exponent vectors for the algebraic integers $a - \alpha b$ and essentially repeat the same algorithm. To arrange for both properties of S to hold simultaneously, well, this would just involve longer exponent vectors having coordinates for all the small prime numbers, for the sign of $a - \alpha b$, for all the "small" primes in $\mathbb{Z}[\alpha]$, and for each unit in the unit basis.

But how are we supposed to do this for a general number n? In fact, how do we even

achieve the first step of finding the polynomial $f(x)$ and the integer m with $f(m) \equiv 0$ mod n? And if we could find it, why should we expect that $\mathbb{Z}[\alpha]$ has all of the nice properties to make Pollard's plan work?

The Number Field Sieve Evolves

There is at the least a very simple device to get started, that is, to find $f(x)$ and m. The trick is to find $f(x)$ *last*. First, one decides on the degree d of f. Next, one lets m be the integer part of $n^{1/d}$. Now write n in the base m, so that $n = m^d + c_{d-1}m^{d-1} + \cdots + c_0$, where the base m "digits" c_i satisfy $0 \le c_i < m$. (If $n > (2d)^d$, then the leading "digit" c_d is 1.) The polynomial $f(x)$ is now staring us in the face; it is $x^d + c_{d-1}x^{d-1} + \cdots + c_0$. So we have a monic polynomial $f(x)$, but is it irreducible?

There are many strategies for factoring primitive polynomials over $\mathbb{Z}$ into irreducible factors. In fact, we have the celebrated polynomial-time algorithm of Arjen Lenstra, Hendrik Lenstra, and Lászlo Lovász for factoring primitive polynomials in $\mathbb{Z}[x]$ (the running time is bounded by a fixed power of the sum of the degree and the number of digits in the coefficients). So suppose we are unlucky and the above procedure leads to a reducible polynomial $f(x)$, say, $f(x) = g(x)h(x)$. Then $n = f(m) = g(m)h(m)$, and from a result of John Brillhart, Michael Filaseta, and Andrew Odlyzko this factorization of n is nontrivial. But our goal is to find a nontrivial factorization of n, so this is hardly unlucky at all! Since almost all polynomials are irreducible, it is much more likely that the construction will let us get started with the number field sieve, and we will not be able to factor n immediately.

There was still the main problem of how one might get around the fact that there is no reason to expect the ring $\mathbb{Z}[\alpha]$ to have any nice properties at all. By 1990 Joe Buhler, Hendrik Lenstra, and I had worked out the remaining difficulties and, incorporating a very practical idea of Len Adleman [1], which simplified some of our constructions,[3] published a description of the *general number field sieve* in [11].

Here is a brief summary of what we did. The norm $N(a - \alpha b)$ (over $\mathbb{Q}$) of $a - \alpha b$ is easily worked out to be $b^d f(a/b)$. This is the homogenized version of f. We define $a - \alpha b$ to be Y-smooth if $N(a - \alpha b)$ is Y-smooth. Since the norm is multiplicative, it follows that if the prod-

uct of various algebraic integers $a - \alpha b$ is a square of an algebraic integer, then so too is the corresponding product of norms a square of an integer. Note too that we know how to find a set of pairs $\langle a, b \rangle$ with the product of $N(a - \alpha b)$ a square. This could be done by using a sieve to discover Y-smooth values of $N(a - \alpha b)$ and then combine them via exponent vector algebra over $\mathbb{F}_2$.

But having the product of the numbers $N(a - \alpha b)$ be a square, while a necessary condition for the product of the $a - \alpha b$ to be a square, is far from sufficient. The principal reason for this is that the norm map takes various prime ideals to the same thing in $\mathbb{Z}$, and so the norm can easily be a square without the argument being a square. For example, the two degree one primes in $\mathbb{Z}[i]$, $2 + i$ and $2 - i$, have norm 5. Their product is 5, which has norm $25 = 5^2$, but $(2 + i)(2 - i) = 5$ is squarefree. (Note that if we are working in the ring of all algebraic integers in $\mathbb{Q}(\alpha)$, then all of the prime ideal factors of $a - \alpha b$ for coprime integers a and b are degree one; namely, their norms are rational primes.) For each prime p let R_p be the set of solutions to $f(x) \equiv 0$ mod p. When we come across a pair $\langle a, b \rangle$ with p dividing $N(a - \alpha b)$, then some prime ideal above p divides $a - \alpha b$. And we can tell which one, since a/b will be congruent modulo p to one of the members of R_p, and this will serve to distinguish the various prime ideals above p. Thus we can arrange for our exponent vectors to have $\#R_p$ coordinates for each prime p and so keep track of the prime ideal factorization of $a - \alpha b$. Note that $\#R_p \le d$, the degree of $f(x)$.

So we have gotten over the principal hurdle, but there are still many obstructions. We are supposed to be working in the ring $\mathbb{Z}[\alpha]$, and this may not be the full ring of algebraic integers. In fact, this ring may not be a Dedekind domain, so we may not even have factorization into prime ideals. And even if we have factorization into prime ideals, the above paragraph merely assures us that the principal ideal generated by the product of the algebraic integers $a - \alpha b$ is the square of some ideal, not necessarily the square of a principal ideal. And even if it is the square of a principal ideal, it may not be a square of an algebraic integer, because of units. (For example, the ideal generated by -9 is the square of an ideal in $\mathbb{Z}$, but -9 is not a square.) And even if the product of the numbers $a - \alpha b$ is a square

---

[3] *Though Adleman's ideas did not change our theoretical complexity estimates for the running time, the simplicities they introduced removed most remaining obstacles to making the method competitive in practice with the quadratic sieve. It is interesting that Adleman himself, like most others around 1990, continued to think of the general number field sieve as purely a speculative method.*

[4] *It is a theorem that if f is a monic irreducible polynomial over $\mathbb{Z}$ with a complex root α and if γ is in the ring of integers of $\mathbb{Q}(\alpha)$, then $f'(\alpha)\gamma$ is in $\mathbb{Z}[\alpha]$. So if γ^2 is a square in the ring of integers of $\mathbb{Q}(\alpha)$, then $f'(\alpha)^2\gamma^2$ is a square in $\mathbb{Z}[\alpha]$.*

of an algebraic integer, how do we know it is the square of an element of $\mathbb{Z}[\alpha]$?

The last obstruction is rather easily handled by using $f'(\alpha)^2$ as a multiplier,[4] but the other obstructions seem difficult. However, there is a simple and ingenious idea of Len Adleman [1] that in one fell swoop overcomes them all. The point is that even though we are being faced with some nasty obstructions, they form, modulo squares, an $\mathbb{F}_2$-vector space of fairly small dimension. So the first thought just might be to ignore the problem. But the dimension is not *that* small. Adleman suggested randomly choosing some quadratic characters and using their values at the numbers $a - \alpha b$ to augment the exponent vectors. (There is one fixed choice of the random quadratic characters made at the start.) So we are arranging for a product of numbers $a - \alpha b$ to not only be a square up to the "obstruction space" but also highly likely actually to be a square. For example, consider the above problem with -9 not being a square. If somehow we cannot "see" the problem with the sign but it sure looks like a square to us because we know that for each prime p the exponent on p in the prime factorization of -9 is even, we might still detect the problem. Here is how: Consider a quadratic character evaluated at -9, in this case the Legendre symbol $(-9/p)$, which is 1 if -9 is a square mod p and -1 if -9 is not a square mod p. Say we try this with $p = 7$. It is easy to compute this symbol, and it turns out to be -1. So -9 is not a square mod 7, and so it cannot be a square in $\mathbb{Z}$. If -9 is a square mod some prime p, however, this does not guarantee it is a square in $\mathbb{Z}$. For example, if we had tried this with 5 instead of 7, then -9 would still be looking like a square. Adleman's idea is to evaluate smooth values of $a - \alpha b$ at the quadratic characters that were chosen and use the linear algebra to create an element with *two* properties: its (unaugmented) exponent vector has all even entries, and its value at each character is 1. This algebraic integer is highly likely, in a heuristic sense, to be a square. If it is not a square, we can continue to use linear algebra over $\mathbb{F}_2$ to create another candidate.

To be sure, there are still difficulties. One of these is the "square root problem". If you have the prime factorizations of various rational integers and their product is a square, you can easily find the square root of the square via its prime factorization. But in $\mathbb{Z}[\alpha]$ the problem does not seem so transparent. Nevertheless, there are devices for solving this too, though it still remains as a computationally interesting step in the algorithm. The interested reader should consult [15].

Perhaps it is not clear why the number field sieve is a good factoring algorithm. A key quantity in a factorization method such as the quadratic sieve or the number field sieve is what I was calling "X" earlier. It is an estimate for the size of the auxiliary numbers that we are hoping to combine into a square. Knowing X gives you the complexity; it is about $\exp(\sqrt{2 \log X \log \log X})$. In the quadratic sieve we have X about $n^{1/2+\varepsilon}$. But in the number field sieve, we may choose the polynomial $f(x)$ and the integer m in such a way that $(a - mb)N(a - \alpha b)$ (the numbers that we hope to find smooth) is bounded by a value of X of the form $\exp(c'(\log n)^{2/3}(\log \log n)^{1/3})$. Thus the number of digits of the auxiliary numbers that we sieve over for smooth values is about the $2/3$ *power* of the number of digits of n, as opposed to the quadratic sieve where the auxiliary numbers have more than half the number of digits of n. That is why the number field sieve is asymptotically so fast in comparison.

I mentioned earlier that the heuristic running time for the number field sieve to factor n is of the form $\exp(c(\log n)^{1/3}(\log \log n)^{2/3})$, but I did not reveal what "c" is. There are actually three values of c depending on which version of the number field sieve is used. The "special" number field sieve, more akin to Pollard's original method and well suited to factoring numbers like $2^{2^9} + 1$ which are near high powers, has $c = (32/9)^{1/3} \approx 1.523$. The "general" number field sieve is the method I sketched in this paper and is for use on any odd composite number that is not a power. It has $c = (64/9)^{1/3} \approx 1.923$. Finally, Don Coppersmith [5] proposed a version of the general number field sieve in which many polynomials are used. The value of "c" for this method is $\frac{1}{3}(92 + 26\sqrt{13})^{1/3} \approx 1.902$. This stands as the champion worst-case factoring method asymptotically. It had been thought that Coppersmith's idea is completely impractical, but [8] considers whether the idea of using several polynomials may have some practical merit.

The State of the Art

In April 1996 a large team (see [7]) finished the factorization of a 130-digit RSA challenge number using the general number field sieve. Thus the gauntlet has finally been passed from the quadratic sieve, which had enjoyed champion status since 1983 for the largest "hard" number factored. Though the real time was about the same as with the quadratic sieve factorization of the 129-digit challenge number two years earlier, it was estimated that the new factorization took only about 15% of the computer time. This discrepancy was due to fewer computers being used on the project and some "down time" while code for the final stages of the algorithm was being written.

The First Twenty Fermat Numbers

m known factorization of $F_m = 2^{2^m} + 1$

0   3
1   5
2   17
3   257
4   65537
5   $641 \cdot P_7$
6   $274177 \cdot P_{14}$
7   $59649589127497217 \cdot P_{22}$
8   $1238926361552897 \cdot P_{62}$
9   $2424833 \cdot 7455602825647884208337395736200454918783366342657 \cdot P_{99}$
10   $45592577 \cdot 6487031809 \cdot 4659775785220018543264560743076778192897 \cdot P_{252}$
11   $319489 \cdot 974849 \cdot 167988556341760475137 \cdot 3560841906445833920513 \cdot P_{564}$
12   $114689 \cdot 26017793 \cdot 63766529 \cdot 190274191361 \cdot 1256132134125569 \cdot C_{1187}$
13   $2710954639361 \cdot 2663848877152141313 \cdot 3603109844542291969 \cdot 319546020820551643220672513 \cdot C_{2391}$
14   C_{4933}
15   $1214251009 \cdot 2327042503868417 \cdot C_{9840}$
16   $825753601 \cdot C_{19720}$
17   $31065037602817 \cdot C_{39444}$
18   $13631489 \cdot C_{78906}$
19   $70525124609 \cdot 646730219521 \cdot C_{157804}$

In the table, the notation P_k means a prime number of k decimal digits, while the notation C_k means a composite number of k decimal digits for which we know no nontrivial factorization.

The history of the factorization of Fermat numbers is a microcosm of the history of factoring. Fermat himself knew about F_0 through F_4, and he conjectured that all of the remaining numbers in the sequence $2^{2^m} + 1$ are prime. However, Euler found the factorization of F_5. It is not too hard to find this factorization, if one uses the result, essentially due to Fermat, that for p to be a prime factor of F_m it is necessary that $p \equiv 1 \bmod 2^{m+2}$, when m is at least 2. Thus the prime factors of F_5 are all 1 mod 128, and the first such prime, which is not a smaller Fermat number, is 641. It is via this idea that F_6 was factored (by Landry in 1880) and that "small" prime factors of many other Fermat numbers have been found, including more than 80 beyond this table.

The Fermat number F_7 was the first success of the Brillhart-Morrison continued fraction factoring method. Brent and Pollard used an adaptation of Pollard's "rho" method to factor F_8. As discussed in the main article, F_9 was factored by the number field sieve. The Fermat numbers F_{10} and F_{11} were factored by Brent using Lenstra's elliptic curve method.

We know that F_{14}, F_{20} and F_{22} are composite, but we do not know any prime factors of these numbers. That they are composite was discovered via *Pepin's criterion*: F_m is prime if and only if $3^{(F_m-1)/2} \equiv -1 \bmod F_m$. The smallest Fermat number for which we do not know if it is prime or composite is F_{24}. It is now thought by many number theorists that every Fermat number after F_4 is composite.

Fermat numbers are connected with an ancient problem of Euclid: for which n is it possible to construct a regular n-gon with straightedge and compass? Gauss showed that a regular n-gon is constructible if and only if $n \geq 3$ and the largest odd factor of n is a product of distinct, prime Fermat numbers. Gauss's theorem, discovered at the age of 19, followed him to his death: a regular 17-gon is etched on his gravestone.

So where is the crossover between the quadratic sieve and the number field sieve? The answer to this depends somewhat on whom you talk to. One thing everyone agrees on: for smaller numbers—say, less than 100 digits—the quadratic sieve is better, and for larger numbers—say, more than 130 digits—the number field sieve is better. One reason a question like this does not have an easy answer is that the issue is highly dependent on fine points in the programming and on the kind of computers used. For example, as reported in [7], the performance of the number field sieve is sensitive to how much memory a computer has. The quadratic sieve is as well, but not to such a large degree.

There is much that was *not* said in this brief survey. An important omission is a discussion of the algorithms and complexity of the linear algebra part of the quadratic sieve and the number field sieve. At the beginning we used Gaussian elimination, as Brillhart and Morrison did with the continued-fraction method. But the size of the problem has kept increasing. Nowadays a factor base of size one million is in the ballpark for record factorizations. Clearly, a linear algebra problem that is one million by one million is not a trifling matter. There is interesting new work

on this that involves adapting iterative methods for dealing with sparse matrices over the real numbers to sparse matrices over $\mathbb{F}_2$. For a recent reference, see [14].

Several variations on the basic idea of the number field sieve show some promise. One can replace the linear expression $a - mb$ used in the number field sieve with $b^k g(a/b)$, where $g(x)$ is an irreducible polynomial over $\mathbb{Z}$ of degree k with $g(m) \equiv 0 \bmod n$. That is, we use two polynomials $f(x)$, $g(x)$ with a common root m mod n (the original scenario has us take $g(x) = x - m$). It is a subject of current research to come up with good strategies for choosing polynomials. Another variation on the usual number field sieve is to replace the polynomial $f(x)$ with a family of polynomials along the lines suggested by Coppersmith. For a description of the number field sieve incorporating both of these ideas, see [8].

The discrete logarithm problem (given a cyclic group with generator g and an element h in the group, find an integer x with $g^x = h$) is also of keen interest in cryptography. As mentioned, Pollard's original idea for the number field sieve was born out of a discrete logarithm algorithm. We have come full circle, since Dan Gordon, Oliver Schirokauer, and Len Adleman have all given variations of the number field sieve that can be used to compute discrete logarithms in multiplicative groups of finite fields. For a recent survey, see [22].

I have said nothing on the subject of primality testing. It is generally much easier to recognize that a number is composite than to factor it. When we use complicated and time-consuming factorization methods on a number, we already know from other tests that it is an odd composite and it is not a power.

I have given scant mention of Hendrik Lenstra's elliptic curve factorization method. This algorithm is much superior to both the quadratic sieve and the number field sieve for all but a thin set of composites, the so-called "hard" numbers, for which we reserve the sieve methods.

There is also a rigorous side to factoring, where researchers try to dispense with heuristics and prove theorems about factorization algorithms. So far we have had much more success proving theorems about probabilistic methods than deterministic methods. We do not seem close to proving that various practical methods, such as the quadratic sieve and the number field sieve, actually work as advertised. It is fortunate that the numbers we are trying to factor have not been informed of this lack of proof!

For further reading I suggest several of the references already mentioned and also [10, 13, 17, 18, 19, 20]. In addition, I am currently writing a book with Richard Crandall, *PRIMES: A computational perspective*, that should be out sometime in 1997.

I hope I have been able to communicate some of the ideas and excitement behind the development of the quadratic sieve and the number field sieve. This development saw an interplay between theoretical complexity estimates and good programming intuition. And neither could have gotten us to where we are now without the other.

Acknowledgments

References

[1] L. M. ADELMAN, *Factoring numbers using singular integers*, Proc. 23rd Annual ACM Sympos. Theory of Computing (STOC), 1991, pp. 64-71.

[2] W. R. ALFORD and C. POMERANCE, *Implementing the self initializing quadratic sieve on a distributed network*, Number Theoretic and Algebraic Methods in Computer Science, Proc. Internat. Moscow Conf., June–July 1993 (A. J. van der Poorten, I. Shparlinski, and H. G. Zimmer, eds.), World Scientific, 1995, pp. 163–174.

[3] J. BRILLHART, D. H. LEHMER, J. L. SELFRIDGE, B. TUCKERMAN, and S. S. WAGSTAFF JR., *Factorizations of $b^n \pm 1$, $b = 2, 3, 5, 6, 7, 10, 11, 12$, up to high powers*, second ed., vol. 22, Contemp. Math., Amer. Math. Soc., Providence, RI, 1988.

[4] E. R. CANFIELD, P. ERDÖS, and C. POMERANCE, *On a problem of Oppenheim concerning "Factorisatio Numerorum"*, J. Number Theory **17** (1983), 1–28.

[5] D. COPPERSMITH, *Modifications to the number field sieve*, J. Cryptology **6** (1993), 169–180.

[6] D. COPPERSMITH, A. M. ODLYZKO, and R. SCHROEPPEL, *Discrete logarithms in $GF(p)$*, Algorithmica **1** (1986), 1–15.

[7] J. COWIE, B. DODSON, R. MARIJE ELKENBRACHT-HUIZING, A. K. LENSTRA, P. L. MONTGOMERY, and J. ZAYER, *A world wide number field sieve factoring record: On to 512 bits*, Advances in Cryptology—Asiacrypt '96, to appear.

[8] M. ELKENBRACHT-HUIZING, *A multiple polynomial general number field sieve*, Algorithmic Number Theory, Second Intern. Sympos., ANTS-II, to appear.

[9] J. Gerver, *Factoring large numbers with a quadratic sieve*, Math. Comp. **41** (1983), 287–294.

[10] A. K. Lenstra, *Integer factoring*, preprint.

[11] A. K. Lenstra and H. W. Lenstra Jr. (eds.), *The development of the number field sieve*, Lecture Notes in Math. vol. 1554, Springer-Verlag, Berlin and Heidelberg, 1993.

[12] A. K. Lenstra and M. S. Manasse, *Factoring by electronic mail*, Advances in Cryptology—Eurocrypt '89 (J.-J. Quisquater and J. Vandewalle, eds.), Springer-Verlag, Berlin and Heidelberg, 1990, pp. 355–371.

[13] H. W. Lenstra Jr., *Elliptic curves and number theoretic algorithms*, Proc. Internat. Congr. Math., Berkeley, CA, 1986, vol. 1 (A. M. Gleason, ed.), Amer. Math. Soc., Providence, RI, 1987, pp. 99–120.

[14] P. L. Montgomery, *A block Lanczos algorithm for finding dependencies over $GF(2)$*, Advances in Cryptology—Eurocrypt '95 (L. C. Guillou and J.-J. Quisquater, eds.), Springer-Verlag, Berlin and Heidelberg, 1995, pp. 106–120.

[15] ____, *Square roots of products of algebraic integers*, Mathematics of Computation 1943–1993, Fifty Years of Computational Mathematics (W. Gautschi, ed.), Proc. Sympos. Appl. Math. vol. **48**, Amer. Math. Soc., Providence, RI, 1994, pp. 567–571.

[16] M. A. Morrison and J. Brillhart, *A method of factorization and the factorization of F_7*, Math. Comp. **29** (1975), 183–205.

[17] A. M. Odlyzko, *The future of integer factorization*, CryptoBytes (The technical newsletter of RSA Laboratories) **1**, no. 2 (1995), 5–12.

[18] C. Pomerance (ed.), *Cryptology and computational number theory*, Proc. Sympos. Appl. Math. vol. **42**, Amer. Math. Soc., Providence, RI, 1990.

[19] ____, *The number field sieve*, Mathematics of Computation 1943–1993, Fifty Years of Computational Mathematics (W. Gautschi, ed.), Proc. Sympos. Appl. Math. vol. **48**, Amer. Math. Soc., Providence, RI, 1994, pp. 465–480.

[20] ____, *On the role of smooth numbers in number theoretic algorithms*, Proc. Internat. Congr. Math., Zürich, Switzerland, 1994, vol. 1 (S. D. Chatterji, ed.), Birkhaüser-Verlag, Basel, 1995, pp. 411–422.

[21] C. Pomerance, J. W. Smith, and R. Tuler, *A pipeline architecture for factoring large integers with the quadratic sieve algorithm*, SIAM J. Comput. **17** (1988), 387–403.

[22] O. Schirokauer, D. Weber, and T. Denny, *Discrete logarithms: The effectiveness of the index calculus method*, Algorithmic Number Theory, Second Intern. Sympos., ANTS-II, to appear.

[23] H. C. Williams and J. O. Shallit, *Factoring integers before computers*, Mathematics of Computation 1943–1993, Fifty Years of Computational Mathematics (W. Gautschi, ed.), Proc. Sympos. Appl. Math. **48**, Amer. Math. Soc., Providence, RI, 1994, pp. 481–531.