Computational Number Theory*

Richard P. Brent
Computing Laboratory
University of Oxford
rpb@comlab.ox.ac.uk

*Presented at ICS'98 Computational Mathematics Workshop, Melbourne, 13 July 1998.

 ICS98wt

Abstract

Number theory provides an infinite supply of computational problems. Some of these are of great practical interest because they are relevant to computer security, authentication, public-key cryptography, etc. For example, a popular public-key algorithm, the RSA algorithm, uses exponentiation modulo a large composite number. To speed up encryption and decryption, the exponentiation should be performed as efficiently as possible. The size of the composite number depends on what level of security is desired. It should not be possible for someone to find the prime factors of this number, either now or in the near future, even using a fast computer with the best known algorithms. In practice this means that the composite number should have at least 150 decimal digits. Alternatives to the RSA algorithm involve the discrete logarithm problem. In this talk I will discuss the integer factorization and discrete logarithm problems and their applications, and outline the current state of algorithms for their solution.

Outline

- Some computational problems in number theory
  - The Riemann hypothesis
  - The Goldbach conjecture
  - The Mertens conjecture
- Number theory in cryptography and authentication
  - Integer factorization and RSA public-key encryption
  - Discrete logarithms and Diffie-Hellman key exchange
  - El Gamal encryption
  - Implementations on smartcards and small computers

Some computational problems in number theory

Many problems in number theory can be solved/disproved/illuminated by numerical computations. Several interesting examples are given in the paper by te Riele and van de Lune [18]. We mention three of them here:

The Riemann hypothesis

The *Riemann hypothesis* (RH) is that all the non-real zeros of the Riemann zeta function

$$\zeta(s) = \sum_{n=1}^{\infty} \frac{1}{n^s}$$

are simple and lie on the "critical line" $\mathcal{R}(s) = \frac{1}{2}$. (The real zeros are at $-2, -4, -6, \ldots$) van de Lune et al [12] have shown that the first 1.5×10^9 zeros lie on the critical line, and Odlyzko has shown that the same is true for many other zeros up to the 10^{20}-th.

The Goldbach conjecture

The *Goldbach conjecture* is that every even positive integer can be written as a sum of two integers which are 1 or an odd prime, e.g.

$$4 = 1 + 3,$$

$$244885595672 = 3163 + \cdots$$

The conjecture has been verified up to 4×10^{11} by Sinisaldo [21].

Weaker conjectures/theorems

Vinogradov *proved* a weaker result: every *sufficiently large* odd integer can be expressed as a sum of three (odd) primes. If we can quantify "sufficiently large" then, in principle, a finite computation can deal with the "small" cases. This has recently been done by te Riele et al on the assumption of the Riemann Hypothesis.

The Mertens conjecture

$$M(x) = \sum_{n=1}^{\lfloor x \rfloor} \mu(n),$$

where $\mu(n)$ is the *Möbius function*,

$$\frac{1}{\zeta(s)} = \sum_{n=1}^{\infty} \frac{\mu(n)}{n^s} \,.$$

The *Mertens conjecture* is that

$$|M(x)| < \sqrt{x}$$

for $x \geq 1$. In support of the conjecture, Lioen and van de Lune verified that

$$|M(x)| < 0.571\sqrt{x} \text{ for } 200 < x \leq 10^{12}.$$

Also, the Riemann hypothesis is equivalent to the statement that

$$(\forall\, \epsilon > 0) \;\; M(x) = O(x^{1/2+\epsilon}) \text{ as } x \to +\infty \,.$$

Disproof of the Mertens conjecture

Odlyzko and te Riele [16] showed, via a computation involving the first 2000 zeros of the Riemann zeta function, that the Mertens conjecture is *false*. Their proof was not constructive, in that it did not give an explicit x which violated the Mertens conjecture. Later, Pintz [17] showed that the conjecture is false for some

$$x < 10^{1.4 \times 10^{64}} .$$

It is probably false for some *much* smaller x, but true for $x < 10^{30}$.

Number theory and cryptography

Until about twenty years ago, number theory was considered to be an esoteric subject with no practical applications. This changed with the introduction of public key cryptography by Diffie and Hellman [7], Merkle and Hellman, and Rivest, Shamir and Adleman (RSA) [20].

In fact, there are many other applications of number theory: see the book by Schroeder [23]. We will consider cryptography (and the related topics of security and authentication) because they involve extensive computations. A good introductory book on cryptography is Schneier [22].

The RSA algorithm

The *RSA algorithm* depends for its security on the (presumed) difficulty of factoring integers which are the product of two (or more) large primes. Here "large" means at least 75 decimal digits (but increasing in accordance with Moore's law).

Suppose $n = pq$ where p and q are distinct (large) primes. Given e relatively prime to $(p-1)(q-1)$, we can use the extended Euclidean algorithm to compute d such that

$$de = 1 \bmod (p-1)(q-1)$$

and thus, by Fermat's little theorem,

$$x^{ed} = x \bmod n \; .$$

The idea of RSA is to encrypt a message x as $y = x^e \bmod n$, and decrypt y as $x = y^d \bmod n$. Anyone who knows x, e and n can perform the encryption, but only someone who knows d can perform the decryption.

RSA continued

To compute d seems to require the factorization of n, and the best known algorithm for this, the *Number Field Sieve* [10], appears to take time

$$\exp\left(c(\log n)^{1/3}(\log\log n)^{2/3}\right) .$$

However, no one has ever *proved* that factorization can not be done in a time polynomial in $\log n$. (It can be done in $O(\log n)^c$ arithmetic operations, but the operations are on very large numbers.)

The related problem of primality testing can be done in polynomial time, using a randomized algorithm, if we are willing to accept a very small probability of error.

Diffie-Hellman key exchange

Suppose two people, Alice and Bob, want to decide on a key K to use in a cryptosystem such as DES, but their telephone line is bugged. Here is how they can do it. Assume that a large prime p and a primitive root g are given in advance (and publicly known).

- Alice chooses a random exponent α, computes $A = g^\alpha \bmod p$, and sends A to Bob.
- Bob chooses a random exponent β, computes $B = g^\beta \bmod p$, and sends B to Alice.
- Now Alice computes $K = B^\alpha \bmod p$ and Bob computes $K = A^\beta \bmod p$.

An eavesdropper (Eve) knows $g^\alpha \bmod p$ and $g^\beta \bmod p$, but how is she to compute $K = g^{\alpha\beta} \bmod n$? This seems to involve solution of the *discrete logarithm problem* (over a finite field): given $g^\alpha \bmod p$, compute $\alpha \bmod p - 1$.

How hard is the discrete log ?

Provided that $p - 1$ has a large prime factor, the discrete logarithm problem over the finite field with p elements appears to be hard. In fact, it is closely related to factorization, as many factorization algorithms have analogues which solve the discrete logarithm problem.

The discrete log problem makes sense in other algebraic contexts, e.g. over an abelian group. Suggestions for cryptography have included finite fields with 2^k elements (but it may be easier to solve the discrete log problem here), and elliptic curves over finite fields.

An *elliptic curve* G over a finite field is a curve of the form

$$y^2 = x^3 + ax + b \bmod p$$

plus a "point at infinity" $\mathcal{O}$. It is easy to define an abelian group on G. In general, it seems difficult to compute discrete logarithms over this group – the best known algorithms take time of order $\sqrt{p}$.

El Gamal encryption

There are public key algorithms based on discrete logarithms, e.g. the El Gamal algorithm [8]. The private key is α. The public key is (p, g, A), where $A = g^\alpha \bmod p$.

To encrypt a message x, choose random k, and compute $(u, v) = (g^k \bmod p, A^k x \bmod p)$.
The encrypted message is (u, v).

To decrypt (u, v), compute $x = v/u^\alpha \bmod p$.

Note that the encrypted message is twice as long as the original message, but in many applications this is unimportant.

Some advantages of El Gamal

- It is not patented (whereas RSA is patented in USA).
- It can be generalized: for example, instead of working over a finite field, we could work over an elliptic curve or other algebraic structure in which it is hard to compute discrete logarithms.

Implementations of RSA in hardware and software

Several of the algorithms which we have mentioned involve the computation of $x^e \bmod n$ where e and n are fixed integers, n is large (typically 512 to 1024 bits), and $0 < x < n$. Thus, we need fast multiple-precision integer arithmetic, including the multiplication and "mod" operations. These may be implemented in software, using several ingenious tricks to speed up the operations (Bernstein, Ford, Karatsuba, Montgomery, ...), or in special-purpose hardware. Schneier's book lists some hardware implementations.

Recently there has been an interest in implementing RSA and similar algorithms on smartcards, which are essentially very small, slow computers with small memories. It is important that common operations, such as checking digital signatures, take no more than a few seconds.

Comparison of RSA and El Gamal

Because RSA (in hardware or software) is slow, it is often combined with a faster cryptosystem such as DES or IDEA. (RSA is used to encrypt the key, which is used for the faster system.)

With numbers of the same size El Gamal is slower than RSA, but because the discrete logarithm problem (especially over elliptic curves) seems more difficult than factorization, we may be able to get away with smaller numbers for the same level of security. Thus, there is an interesting tradeoff, and it is not clear which algorithm is best.

Conclusions

- Number theory provides many interesting and computationally intensive problems.
- Applications to cryptography require fast computation of multiple-precision integer multiplication and division.

References

[1] A. O. L. Atkin and F. Morain, Elliptic curves and primality proving, *Math. Comp.* **61** (1993), 29–68. Programs available from `ftp://ftp.inria.fr/INRIA/ecpp.V3.4.1.tar.Z` .

[2] D. J. Bernstein, Multidigit modular multiplication with the explicit Chinese remainder theorem, available from `http://pobox.com/~djb/papers.html` .

[3] R. P. Brent, *Large factors found by ECM,* Computer Sciences Laboratory, Australian National University, Dec. 1995 (and more recent updates). `ftp://nimbus.anu.edu.au/pub/Brent/champs.ecm` .

[4] R. P. Brent, *Factorization of the tenth and eleventh Fermat numbers,* Report TR-CS-96-02, CS Lab., ANU, Feb. 1996, 25 pp. `ftp://nimbus.anu.edu.au:/pub/Brent/rpb161tr.dvi.gz` . Revision to appear in *Math. Comp.*, 1998.

[5] 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*, 2nd ed., Amer. Math. Soc., Providence, RI, 1988.

[6] C. Caldwell, The Dubner PC Cruncher – a microcomputer coprocessor card for doing integer arithmetic, review in *J. Rec. Math.* **25** (1), 1993.

[7] W. Diffie and M. E. Hellman, New directions in cryptography, *IEEE Trans. on Information Theory* **IT-22** (1976), 644–654.

[8] T. El Gamal, A public-key cryptosystem and a signature scheme based on discrete logarithms, *IEEE Trans. on Information Theory* **IT-31** (1985), 469–472.

[9] A. Karatsuba and Y. Ofman, Multiplication of multidigit numbers on automata, *Soviet Phys. Dokl.* **7** (1962), 595–596.

[10] A. K. Lenstra and H. W. Lenstra, Jr. (editors), *The development of the number field sieve,* Lecture Notes in Mathematics **1554**, Springer-Verlag, Berlin, 1993.

[11] H. W. Lenstra, Jr., Factoring integers with elliptic curves, *Annals of Mathematics* (2) **126** (1987), 649–673.

[12] J. van de Lune, H. J. J. te Riele and D. T. Winter, On the zeros of the Riemann zeta function in the critical strip, IV, *Math. Comp.* **46**, 667–681.

[13] P. L. Montgomery, Speeding the Pollard and elliptic curve methods of factorization, *Math. Comp.* **48** (1987), 243–264.

[14] P. L. Montgomery, *An FFT extension of the elliptic curve method of factorization,* Ph. D. dissertation, Mathematics, University of California at Los Angeles, 1992.

[15] P. L. Montgomery, A survey of modern integer factorization algorithms, *CWI Quarterly* **7** (1994), 337–366.

[16] A. M. Odlyzko and H. J. J. te Riele, Disproof of the Mertens conjecture, *J. reine angew. Math.* **357** (1985), 138–160.

[17] J. Pintz, An effective disproof of the Mertens conjecture, *Astérisque* **147/148** (1987), 325–333.

[18] H. J. J. te Riele and J. van de Lune, Computational number theory at CWI in 1970–1994, *CWI Quarterly* **7** (1994), 285–335.

[19] H. Riesel, *Prime Numbers and Computer Methods for Factorization*, second edition, Birkhäuser, Boston, 1994.

[20] R. L. Rivest, A. Shamir and L. Adleman, A method for obtaining digital signatures and public-key cryptosystems, *Comm. ACM* **21** (1978), 120–126.

[21] M. K. Sinisaldo, Checking the Goldbach conjecture up to 4×10^{11}, *Math. Comp.* **61**, 931–934.

[22] B. Schneier, *Applied Cryptography: Protocols, Algorithms, and Source Code in C*, second edition, John Wiley and Sons, 1996.

[23] M. R. Schroeder, *Number Theory in Science and Communication: With Applications in Cryptography, Physics, Digital Information, Computing, and Self-Similarity*, third edition, Springer-Verlag, 1997.