A SUBEXPONENTIAL ALGORITHM FOR THE DISCRETE LOGARITHM PROBLEM WITH APPLICATIONS TO CRYPTOGRAPHY

(abstract)

Leonard Adleman*
Department of Mathematics
and
Laboratory for Computer Science
Massachusetts Institute of Technology

I. Introduction

In 1870 Bouniakowsky [2] published an algorithm to solve the congruence $a^x \equiv b \text{ MOD } (q)$. While his algorithm contained several clever ideas useful for small numbers, its asymptotic complexity was $O(q)$. Despite its long history, no fast algorithm has ever emerged for the Discrete Logarithm Problem and the best published method, due to Shanks [10] requires $O(q^{1/2})$ in time and space. The problem has attracted renewed interest in recent years because of its use in cryptography [7], [15],[19]. In particular, the security of the Diffie-Hellman Public Key Distribution System [7] "depends crucially on the difficulty of computing logarithms MOD q". We present a new algorithm for this problem which runs in RTIME better than $O(q^{\varepsilon})$ for all $\varepsilon > 0$.† While no effort is made to present the most efficient incarnation of

---

*Research sponsored by National Science Foundation Grant MCS78-04343.

†Actually our algorithm runs in RTIME $O(2^{(O(\sqrt{\log(q)\log\log(q)}))})$. RTIME denotes Random Time and refers to algorithms which may use random numbers in their processing. For example, the well known composite testing algorithms of Solovay & Strassen [21], Miller [11] and Rabin [16] run in RTIME $(O(\log^3(q)))$. For precise definitions see [1], [11] and [9].

CH1471-2/79/0000-0055$00.75 © 1979 IEEE

the new algorithm, it is clearly practical for numbers of moderate size and this has consequences for the Diffie-Hellman scheme. We will discuss these consequences in Section III.

II. The Method

In 1971 Morrison and Brillhart [12] introduced a new heuristic for factoring and used it to factor F_7 $(2^{2^7}+1)$. While hundreds of heuristics for factoring have been proposed in the past, most have applications only in special cases, and have poor worst case behaviors. Even those which seem to have good running times, often rely on assumptions which are currently unprovable. The Morrison-Brillhart method is different. Not only is it (or its refinement due to Schroeppel) the fastest algorithm in practice, but it is based on provable principles. Dixon [8] has recently done an analysis of the method, and his results can be seen to show that it works in RTIME $(O(2^{O(\sqrt{\log(q)\log\log(q)})}))$.

The Morrison-Brillhart scheme makes use of the long known result that a disproportionately large portion of numbers not only have a small prime factor but are entirely composed of small prime factors. We will base our discrete logarithms algorithm on the same idea:

Definition For all $a,b \in \mathbb{N}$, a is smooth[†] with respect to the bound b $\Leftrightarrow$ $(\forall p \in \mathbb{N})$ $[[p \in \text{primes} \ \& \ p|a] \Rightarrow p < b]$.

Notice that if a is smooth with respect to b then $a = \prod_{i=1}^{K} p_i^{e_i}$ where $p_1,p_2,\ldots,p_K$ are the primes less than b. Thus for each smooth a we have an associated vector $\langle e_1,e_2,\ldots,e_K\rangle$. We will denote this vector $\vec{a}$.

Below we present a restricted form of the algorithm which works when q is prime and a,b are generators (this is a typical case for cryptographic use). We will then show how to use the algorithm to handle the general case. The constant c in the algorithm is probably small.

"on input a,b,q:

1. (Using Morrison-Brillhart) Factor $q-1 = p_1^{e_1} p_2^{e_2} \cdots p_z^{e_z}$
2. For each $p_\ell^{e_\ell} | q$ proceed in steps until m_ℓ is obtained:

[†]The term smooth for such numbers was suggested by R. Rivest.

STEP i:

(a) (By guessing and checking) find r_i, s_i such that a^{r_i} MOD (q) and ba^{s_i} MOD (q) are smooth with respect to the bound $2^{c\sqrt{\log(q)\log\log(q)}}$. (In more syntactical terms, what we are seeking are numbers all of whose prime factors have length about the square root of the length of q)

(b) (Using Gaussian elimination) check if over $\mathbb{Z}_{p_\ell^{e_\ell}}$

ba^{s_i} MOD $(\vec{q})$ is dependent on

$\{a^{r_1}$ MOD $(\vec{q}), \ldots, a^{r_i}$ MOD $(\vec{q})\}$. If yes, calculate α_j's such that

$$ba^{s_i} \text{ MOD } (\vec{q}) \equiv \left(\sum_{j=1}^{i} \alpha_j a^{r_i} \text{ MOD } (\vec{q}) \right) \text{ MOD } (p_\ell^{e_\ell})$$

$$\text{then } m_\ell = \left(\sum_{j=1}^{i} \alpha_j r_j \right) \text{ MOD } (p_\ell^{e_\ell}) - s_i$$

3. (Using Chinese Remainder Theorem)

calculate and output x such that $x \equiv m_\ell$ MOD $(p_\ell^{e_\ell})$ $\ell = 1,2,\ldots,z$ "

To handle the cases where q is not prime or a or b are not generators do the following:

(I) If q is prime (checkable in our time bound) and a and b are not generators (checkable in our time bound) then

a) guess a generator g (these are abundant)

b) solve (using II below) $g^x \equiv a$ MOD (q), $g^y \equiv b$ MOD (q)

c) calculate w = GCD(x,y,q-1), calculate z such that $(\frac{x}{w})z \equiv (\frac{y}{w})$MOD(q-1) then $a^z \equiv b$ MOD (q) (if such a z exists).

(II) If q is prime a is a generator but b is not then

1. find (by guessing & checking) a random K such that $b' \equiv ba^K$MOD(q) is a generator

2. use the original algorithm to find z such that $a^z \equiv b'$MOD(q). then $a^{z-K} \equiv b$ MOD (q)

(III) If q is composite then factor q (using Morrison-Brillhart) and solve modulo each prime power divisor. Reconcile the results using Chinese Remainder Theorem.

III. Applications to Cryptography

The Diffie-Hellman Public Key Distribution System [7] is used as follows:

1. A prime q and generator g are made publicly available
2. When A wishes to communicate with B:
   a) A chooses a random number a and sends $g^a MOD(q)$ to B
   b) B chooses a random number b and sends $g^b MOD(q)$ to A
   c) A & B both compute $g^{ab} MOD(q)$ and use this as a key for future communications.

A tapper has g, q, $g^a MOD(q)$, $g^b MOD(q)$ so if he can take discrete logs he can calculate $g^{ab} MOD(q)$ and compromise communications.

As Diffie & Hellman state:

> "The security of our system depends crucially on the difficulty of computing logarithms modq.... For now we assume the best known algorithm for computing logs mod q is in fact close to optimal and hence that $q^{\frac{1}{2}}$ is a good measure of the problems complexity."

Diffie & Hellman then argue that if q is chosen to be approximately 200 bits long then their system would be secure. In fact the best known algorithm (Shanks') would have required approximately 3×10^{16} years on a 1 microsecond per operation machine to crack the system. We think it prudent to assume that our new algorithm would run in $e^{\sqrt{\log_e(q)\log_e\log_e(q)}}$. We base this assumption on arguments made by Schroeppel in reference to his (already running) refinement of the Morrison-Brillhart factoring algorithm [20]. Accordingly, our new algorithm could be expected to compromise a system based on a 200 bit q in 2.6 days on a 1 microsecond per operation machine. This is not to suggest that the Diffie-Hellman system is insecure, rather that larger moduli are required (see [18]).

IV. Proof of Correctness (outline)

(i) We first argue that the output x is $\log_a(b)$. It is enough to show that $m_\ell \equiv \log_a(b) MOD(p_\ell^{e_\ell})$ $\ell = 1,2,\ldots,z$. To see this consider $y \equiv \log_a(b) - m_\ell MOD(q-1)$.

then

a) $a^y a^{m_\ell} \equiv b \; MOD\,(q)$ and

b) $a^y \equiv a^{(-\Sigma\alpha_j r_j) MOD(p_\ell^{e_\ell})} a^{s_i} b \; MOD\,(q)$

but the $a^{r_j} MOD\,(q)$ and $a^{s_i} b \; MOD\,(q)$ are smooth, and the α_j's were calculated

to insure that

c) $a^{(-\Sigma\alpha_j r_j) \text{MOD}(p_\ell^{e_\ell})} a^{s_i} b = d_1^{f_1} d_2^{f_2} \ldots d_w^{f_w}$

where $d_1 \ldots d_w$ are the primes less than bound $2^{c\sqrt{\log(q)\log\log(q)}}$ and $f_K \equiv 0 \text{ MOD } (p_\ell^{e_\ell})$ $K = 1,\ldots,w$.

It follows from b) and c) that a^y is a $p_\ell^{e_\ell}$th residue MOD(q) and thus $y \equiv 0 \text{ MOD } (p_\ell^{e_\ell})$.

(ii) That there are sufficiently many r_i and s_i for which $a^{r_i} \text{MOD}(q)$ and $a^{s_i} b \text{ MOD } (q)$ are smooth with respect to $2^{c\sqrt{\log(q)\log\log(q)}}$ follows from deep number theoretic results [13][17][4][5][6][3] concerning the function $\psi(x,y)$ = the number of numbers less than x all of whose prime factors are less than y. The analysis needed for this algorithm is carried out in Dixon [8].

(iii) To see that we will with high probability produce an s_i such that $ba^{s_i} \text{MOD}(q)$ is dependent on $\{a^{r_1} \text{MOD}(q),\ldots,a^{r_i} \text{MOD}(q)\}$, consider the following argument. If in step i we replace $ba^{s_i} \text{MOD}(q)$ with $a^{r_i} \text{MOD}(q)$, then after $\pi(2^{c\sqrt{\log(q)\log\log(q)}}) + y$ steps we would have had at least y dependencies (since the dimension of the space is $\pi(2^{c\sqrt{\log(q)\log\log(q)}})$). However, at any particular stage, $a^{s_i} \text{MOD}(q)$ is a random choice from the set $\{1,2,\ldots,q-1\}$, and so is $a^{r_i} b \text{ MOD } (q)$. It follows that at any particular stage there is equal probability that $a^{r_i} \text{MOD}(q)$ or $a^{s_i} b \text{ MOD } (q)$ will be dependent. Thus the odds that a dependent $a^{s_i} b \text{ MOD } (q)$ will not have been found are smaller than $1/2^y$.

(iv) When $e_\ell > 1$ then $\mathbb{Z}_{p_\ell^{e_\ell}}$ is not a field and the theory of vector spaces implicit in the use of Gaussian Elimination is no longer directly applicable. However, much of the theory extends to the case where we are working over a ring rather than a field (module theory) and no problems arise. See [14].

Acknowledgements

Thanks to Ron Rivest and Adi Shamir for their contributions to this paper. Also to the participants in this year's Diophantine Complexity seminar at M.I.T. for suggestions which helped clarify the proof.

The author would also like to acknowledge Ralph Merkle [14] who has worked on this problem and independently produced several of the key ideas. Also Nick Pippenger for suggesting the use of the theory of modules.

References

[1] Adleman, L. and Manders, K., "Reducibility, Randomness, and Intractibility", Proc. 9th Annual ACM Symp. on Theory of Computing (1977), 151-163.

[2] Bouniakowsky, V., Bull. Ac. Sc. St. Petersburg 14,(1870) 356-375.

[3] Buchstab, A.A., "On Those Numbers in an Arithmetic Progression All Prime Factors of Which Are Small in Order of Magnitude", Dokl. Akad. Nauk SSR (N.S.), Vol. 67, pp. 5-8 (1949) (Russian).

[4] de Bruijn, N.G., "On the Number of Positive Integers $\leq x$ and Free of Prime Factors $> y$", Nederl. Akad. Wetensch. Proc. Ser. A., 54, 50-60 (1951).

[5] de Bruijn, N.G., "On the Number of Positive Integers $\leq x$ and Free of Prime Factors $> y$, II", Nederl. Akad. Wetensch. Proc. Ser. A, 69, 239-247 (1966).

[6] de Bruijn, N.G., "The Asymptotic Behavior of a Function Occurring in the Theory of Primes", J. Indian Math. Soc. (N.S.), 15, 25-32 (1951).

[7] Diffie, W. and Hellman, M., "New Directions in Cryptography", IEEE Transactions on Information Theory, Vol. IT-22, November 1976.

[8] Dixon, J.D., "Asymptotically Fast Factorizations of Integers", to appear.

[9] Gill, J., "Computational Complexity of Probabilistic Turing Machines", Conf. Records 6th ACM Symposium on Theory of Computing", 91-95 (1974).

[10] Knuth, D., "The Art of Computer Programming", vol. 3, Addison Wesley, p. 9 (1973).

[11] Miller, G.L., "Riemann's Hypothesis and Tests for Primality", Ph.D. Thesis, Berkeley (1975).

[12] Morrison, M. and Brillhart, J., "A Method of Factoring and the Factorization of F_7", Math. Comp., v. 29, 129, pp. 183-205.

[13] Norton, K.K., "Numbers with Small Prime Factors, and the Least k-th Power Non-Residue", Memoirs Amer. Math. Soc., 106, 9-27 (1971).

[14] Pohlig, S.C., "Algebraic & Combinatoric Aspects of Cryptography", Stanford University, Stanford Electronics Laboratory, Technical Report No. 6602-1 (1977).

[15] Pohlig, S.C. and Hellman, M.E., "An Improved Algorithm for Computing Logarithms Over GF(p) and Its Cryptographic Significance", IEEE Trans. on Information Theory, Vol. IT-24, No. 1, January 1978.

[16] Rabin, M.O., "Probabilistic Algorithms", in Algorithms and Complexity, New Directions and Recent Results, J. Traub (ed.), Academic Press, 21-40.

[17] Rankin, R.A., "The Difference Between Consecutive Prime Numbers", J. London Math. Soc., 13, 242-247 (1938).

[18] Rivest, R., Shamir, A., Adleman, L., "A Method for Obtaining Digital Signatures and Public Key Cryptosystems", Communications of the ACM, Feb. 1978.

[19] Schlotz, R.A. and Welch, L.R., "Generalized Residue Sequences", Proc. Int. Conf. Comm. Seattle, WA, June 1973.

[20] Schroeppel, R., Personal communications to R. Rivest.

[21] Strassen, V. and Solvay, R., "Fast Monte-Carlo Tests for Primality", SIAM Journal on Computing (1977), 84-85.