Monte Carlo Methods for Index Computation $(\operatorname{mod} p)$
Author(s): J. M. Pollard
Source: *Mathematics of Computation*, Vol. 32, No. 143 (Jul., 1978), pp. 918-924
Published by: American Mathematical Society
Stable URL: http://www.jstor.org/stable/2006496
Accessed: 16-06-2016 10:27 UTC

MATHEMATICS OF COMPUTATION, VOLUME 32, NUMBER 143
JULY 1978, PAGES 918–924

Monte Carlo Methods for Index Computation (mod p)

By J. M. Pollard

Abstract. We describe some novel methods to compute the index of any integer relative to a given primitive root of a prime p. Our first method avoids the use of stored tables and apparently requires $O(p^{1/2})$ operations. Our second algorithm, which may be regarded as a method of catching kangaroos, is applicable when the index is known to lie in a certain interval; it requires $O(w^{1/2})$ operations for an interval of width w, but does not have complete certainty of success. It has several possible areas of application, including the factorization of integers.

1. A Rho Method for Index Computation. The concept of a random mapping of a finite set is used by Knuth [1, pp. 7–8] to explain the behavior of a type of random number generator. A sequence obtained by iterating such a function in a set of p elements is 'rho-shaped' with a tail and cycle which are random variables with expectation close to

$$\sqrt{(\pi p/8)} \doteq 0.6267\sqrt{p}, \tag{1}$$

(as shown first in [2], [3]). Recently [4], we proposed that this theory be applied to recurrence relations such as

$$x_{i+1} \equiv x_i^2 \pm 1 \pmod{p}, \tag{2}$$

and showed how a very simple factorization method results, in which a prime factor p of a number can be found in only $O(p^{1/2})$ operations. The method has been further discussed by Guy [5] and Devitt [6], who have found it suitable for use in programmable calculators.

We now suggest that the same theory can be applied to sequences such as $x_0 = 1$,

$$x_{i+1} \equiv \begin{Bmatrix} qx_i \\ x_i^2 \\ rx_i \end{Bmatrix} \pmod{p} \quad \text{for} \begin{Bmatrix} 0 < x_i < \frac{1}{3}p \\ \frac{1}{3}p < x_i < \frac{2}{3}p \\ \frac{2}{3}p < x_i < p \end{Bmatrix}, \tag{3}$$

where r is a primitive root of the prime p, q is any integer, and x_i is always taken in the range $0 < x_i < p$. The idea of this definition, which can be varied in many ways, is that the three possibilities are chosen in a 'random' manner, and the resulting sequence is sufficiently 'complicated' to be regarded as a random mapping; in addition, all the x_i are easily expressible in terms of q and r. As a consequence, we can give an

Received May 1, 1977; revised November 18, 1977.
AMS (MOS) subject classifications (1970). Primary 10–04, 10A10, 12C99.
Key words and phrases. Indices, primitive roots, finite fields.

algorithm to compute the index of q in $O(p^{1/2})$ operations, and with a very small storage requirement.

The method is an alternative to the following method of D. Shanks (see [7, pp. 9, 575–576]). Put $m = [\sqrt{p}] + 1$, and rewrite the equation $q \equiv r^{am+b} \pmod p$, $(0 \leqslant a, b < m)$ as $qr^{-b} \equiv r^{am} \pmod p$. To solve this, compute the sets qr^{-b} and $r^{am} \pmod p$, and find a common member by sorting both sets (the idea has other applications [8], [9]). The method is of order $p^{1/2}\log p$ and requires storage $O(p^{1/2})$.

The main interest of our method, which may be slightly faster, is that it shows that such storage is unnecessary. We are not aware of any particular need for such index calculations, but believe that the ideas may have other applications (such as those described in the last section).

Diffie and Hellman [10] conjecture, and hope, that the estimate $O(p^{1/2})$ is the best possible (for a general prime). But there is some possibility of obtaining a more powerful method from the ideas of Western and Miller, to which Miller [11] has recently drawn attention; we sketch a possible approach. We generate the sequence $r^i \pmod p$ and (for $r^i > p$) try to find numbers which factor entirely into primes below some limit (as Brillhart and Morrison [12] do with their Q_i)—or perhaps primes whose epacts [4], [5] are below some limit, the factoring being by our method mentioned earlier. After a sufficient number of successes, we compute the indices of these primes as the solution of a set of linear equations (mod $p-1$). Then, to obtain the index of an arbitrary q, we need only find one number $qr^i \pmod p$ which factors into this set of primes.

Continuing the description of our method based on (3), we define sequences (a_i) and (b_i) such that

$$x_i \equiv q^{a_i} r^{b_i} \pmod p. \tag{4}$$

Thus, we set $a_0 = 0$ and $a_{i+1} \equiv a_i + 1, 2a_i$, or $a_i \pmod{p-1}$, according to the three cases in (3); similarly, we put $b_0 = 0$ and $b_{i+1} \equiv b_i$, $2b_i$ or $b_i + 1 \pmod{p-1}$.

We introduce an idea of R. W. Floyd [1, p. 4] which was used in [4]; we will have $x_i = x_{2i}$ just when i is a positive multiple of the cycle length and not less than the tail length. The least such i has been named the *epact* [5]. For a true random mapping, it has expectation close to

$$\sqrt{(\pi^5 p/288)} \doteq 1.0308\sqrt{p},$$

and we conjectured [4] that this holds also for sequences of type (2). For (3), we believe that the constant may be different, but not by much. Thus, a calculation on 50 primes near 10^4 gave a mean value for $e(p)/p^{1/2}$ of 1.08. The individual epacts are quite variable, some being as large as $3\sqrt{p}$, as they are for the epacts associated with (2) for which Guy [5] conjectures that

$$\max_{p \leqslant x} e(p) \sim (x \ln x)^{1/2} \quad \text{as } x \longrightarrow \infty.$$

Our method is to run through the sets

$$(x_i, a_i, b_i, x_{2i}, a_{2i}, b_{2i}), \qquad i = 1, 2, 3, \ldots, \tag{5}$$

generating each from the previous one, until the epact is reached, i.e. $x_i = x_{2i}$. Then from (4) we have an equation

$$q^m \equiv r^n \pmod{p}, \tag{6}$$

where $m \equiv a_e - a_{2e}$ and $n \equiv b_{2e} - b_e \pmod{p-1}$. In most cases, this substantially computes the index of q.

Thus, let us compute $d = \text{g.c.d.}(m, p-1)$ by the extended form of Euclid's algorithm [1, p. 302] obtaining an equation $d = \lambda m + \mu(p-1)$. Raising (6) to the power λ gives $q^d \equiv r^{\lambda n} \pmod{p}$, where λn is of form dk, and so

$$q \equiv r^k \theta^i \pmod{p}, \tag{7}$$

where $\theta \equiv r^{(p-1)/d}$ is a dth root of unity, and i $(0 \leqslant i \leqslant d-1)$ remains to be determined. For this we recommend trial of the possible values after computing r^k and θ by the usual power algorithm.

This assumes that d is small, and this point may need amplification. The hardest case for computing indices is when $p - 1 = 2p'$, p' prime; otherwise, we can use the factorization of $p-1$ to obtain the index in several stages, as will be described. In the hard case it is almost certain that $d = 1$ or 2. For a general prime, we argue that the probability that any prime factor p' of $p-1$ divides m, and therefore d, is only $1/p'$; this leads us to predict that values of d comparable with $p^{1/2}$, say, will still be extremely rare.

Example 1. $p = 999959$, $r = 7$, $q = 3$. Here $p - 1 = 2p'$, p' prime, so that 7, being a quadratic nonresidue, is also a primitive root. An ICL 1906S computer takes about 0.5 sec. to find that $e = 1174$, $x_e = x_{2e} = 11400$, $m = 310686$ and $n = 764000$. Thus,

$$3^{310686} \equiv 7^{764000} \pmod{p}.$$

Euclid's algorithm gives $2 = 148845 \times 310686 - 46246 \times 999958$, from which we deduce that $3^2 \equiv 7^{356324}$ and $3 \equiv \pm 7^{178162} \pmod{p}$. Since 3 is a quadratic residue and -1 a nonresidue, the plus sign holds, and the index is 178162.

The algorithm could be used in many programmable calculators. For the very smallest presently available (such as the HP-25) the method is still just possible, but storage limitations, for data and program, force us to break up the calculation into several parts, performed by separate programs.

Example 2. $p = 99989$, $r = 2$, $q = 107$. The first program runs through the pairs (x_i, x_{2i}) searching for the epact; the other variables are absent. The HP-25 takes 35 minutes to find that $e = 357$, $x_e = x_{2e} = 60609$. The second program uses a single set of three related variables (x_i, a_i, b_i). By running through these sets as far as $i = e$, the program obtains $a_e = 91377$, $b_e = 74146$. It then runs on until the first repetition of x_e, which will occur after a further c steps (where c, the cycle length, is a factor of e). In our case, $c = 119$ and $a_{c+e} = 4749$, $b_{c+e} = 84434$; the second program takes a further 15 minutes. Hence we have

$$107^{86628} \equiv 2^{10288} \pmod{p}.$$

A third program applies Euclid's algorithm, obtaining $4 = -12027 \times 86628 + 10420 \times 99988$, from which we have $107^4 \equiv 2^{51368} \pmod{p}$. Therefore

$$107 \equiv 2^{12842}\theta^i,$$

where $\theta \equiv 2^{(p-1)/4} \equiv 2^{24997}$ and $0 \leqslant i \leqslant 3$. With the help of a fourth program, for computing powers (mod p), we find that $i = 3$, and the index of 107 is 87833. The third and fourth programs take less than a minute. Here, as in Example 1, we have chosen a case where the epact turned out to be close to $p^{1/2}$.

2. A Multistage Method, and Other Variants. By making use of the factorization of $p - 1$, we can improve our method to be of order $p_1^{1/2}$, where p_1 is the greatest prime factor of $p - 1$; this can equally be done for Shanks' method, making it of order $p_1^{1/2} \log p$ (a method of order p_1 is due to R. I. Silver; see next revision of [7]). To illustrate, we assume that $p - 1 = st$ and compute the index of q in two stages. First, we apply our existing method with $Q \equiv q^s$ and $R \equiv r^s \pmod{p}$ replacing q and r. This leads to an equation $Q^M \equiv R^N \pmod{p}$, or $q^{sM} \equiv r^{sN} \pmod{p}$. From this we obtain (7) with d a multiple of s. Now we have to express qr^{-k} as a power of θ, and can use the same method again with $Q' \equiv qr^{-k}$ and $\theta \equiv r^{(p-1)/d}$ for q and r.

Example 3. We take $p = 99989$, $r = 2$, $q = 107$, as in Example 2, but note that $p - 1 = 2^2.7.3571$. First, we compute $Q \equiv q^{28} \equiv 11908$, $R \equiv r^{28} \equiv 64980$. Then, following the method of Example 2, we get $e = 88$, $x_e = x_{2e} = 91305$ (8 minutes), then $a_e = 8288$, $b_e = 11665$, $c = 22$, $a_{e+c} = 61152$, $b_{e+c} = 93549$ (4 minutes more). Hence

$$Q^{47124} \equiv R^{81884} \pmod{p},$$

from which $q^{19628} \equiv r^{93016}$. Next, $d = 28$ and $28 = 1569 \times 19628 - 308 \times 99988$, so $q^{28} \equiv r^{59612}$ and $q \equiv r^{2129}\theta^i$, where $\theta \equiv r^{3571}$ and $0 \leqslant i \leqslant 27$. Trying $i = 0$, 1, 2, . . . in turn, we get $r^{2129}\theta^{10} \equiv -q$, hence the index of q is $2129 + 10 \times 3571 + \frac{1}{2}.99988 = 87833$.

Of course, we are free to use this method even in the hard case $p - 1 = 2p'$, taking $s = 2$. Thus, we could write q^2 and r^2 in (3) in place of q and r; but we preferred not to do so in our original description.

Many other variations in (3) are possible. Thus, we could give a different decision rule between the same three alternatives. We could manage with only *two* of these, say rx_i and x_i^2; then we find that $2^k | m$ in (6) if $2^k | p - 1$, which may be a nuisance in the single stage algorithm. It would be convenient for programming, especially if q and r are small, to use only rx_i and qx_i. But this could be disastrous–if ind q is small and positive, the epact would be of order p instead of $p^{1/2}$ (thus, with $q \equiv r^2 \pmod{p}$, we have $e \geqslant c > \frac{1}{2}p$).

We would not expect any difficulty in translating our algorithms to an arbitrary finite field; but we have not performed any experiments.

3. A Lambda Method for Catching Kangaroos, and Some Applications. A more fundamental change is to a method resembling a lambda rather than a rho. We illustrate with the problem of catching a kangaroo which is travelling along a known path in a

series of what appear to be unpredictable bounds; in reality, their lengths are a function of the state of the ground at the point of take-off. We suppose that this function takes values at random from an integer set S, of mean m and largest member L (known as the upper bound).

We require the services of a second tame kangaroo T of identical jumping behavior. We cause T to start at some point x_0, and take N bounds, arriving at x_N. A hole dug at this point, and suitably camouflaged, will catch the wild kangaroo W if he lands on any of the points $x_0, x_1, \ldots, x_N$. For a crude estimate, we can say that W has N (or $N+1$) chances of losing his freedom with independent probabilities $1/m$. Thus, if $N = \theta m$, he will be caught with probability about

$$1 - (1 - 1/m)^{\theta m} \doteq 1 - e^{-\theta} \quad \text{(for large } m\text{)},$$

i.e. 0.63, 0.86, 0.95, 0.98 for $\theta = 1, 2, 3, 4$. In case of success (for us) a comparison of the distance recorders carried by W and T enables us to find W's starting position, which is what we really wanted, and W can go free. Note that young kangaroos will rarely be caught in this kind of trap, which is good.

As an application, we describe a method to compute the index of an integer $q \pmod p$ given that it lies in some range $A < \text{ind } q < B$. The method is of order $w^{1/2}$, where $w = B - A$, as compared with $w^{1/2} \log w$ for the obvious modification of the method of [7] (we consider the hard case $p - 1 = 2p'$).

Our kangaroos are represented by sequences of the form

$$x_{i+1} \equiv x_i \times r^{f(x_i)} \pmod p,$$

where the 'random' function $f(x)$ takes values in the set S; the distance from x_0 to x_i is then

$$d_i = f(x_0) + f(x_1) + \cdots + f(x_{i-1}).$$

First T starts at $x_0 \equiv r^B \pmod p$ and travels to x_N, a distance of d_N. Then W starts at $x'_0 \equiv q \equiv r^{\text{ind } q} \pmod p$. Capture is indicated by $x'_M \equiv x_N \pmod p$, after which we calculate $\text{ind } q \equiv B + d_N - d'_M \pmod{p-1}$; otherwise W can stop when d_M exceeds $B + d_N - A$, since he has then certainly passed the trap.

We give an approximate analysis in which the nature of the set S still does not enter, other than through its mean m. Let $m = \alpha w^{1/2}$ and, as before, $N = \theta m = \alpha\theta w^{1/2}$. The total work of the algorithm when successful is $N + M$, with expected value

$$N + \left(\frac{1}{2} \cdot \frac{w}{m} + N\right) = w^{1/2}\left(2\alpha\theta + \frac{1}{2\alpha}\right); \tag{8}$$

this has minimum $2\theta^{1/2}w^{1/2}$ at $\alpha = 1/2\theta^{1/2}$, e.g. for $\theta = 4$, $\alpha = \frac{1}{4}$, $m = \frac{1}{4}w^{1/2}$, $N = w^{1/2}$. This represents the best choice of α if the probability of failure is small (i.e. θ is large).

We propose to compute and store all the powers $r^s \pmod p$, $(s \in S)$ required by the algorithm, and hence require S to be sparse (certainly $|S| \ll w^{1/2}$, or our method is pointless). In the following example, we could allow only four values.

Example 4. $p = 99989$, $r = 8$, $q = 428$; we are told that $-5000 < \text{ind } q < 0$

(in fact, ind q was known from Example 2). Our HP-25 program takes $f(x_i) = 4^j$ $(0 \leqslant j \leqslant 3)$, where $j \equiv x_i \pmod 4$; thus the numbers r, r^4, r^{16} and r^{64} (mod p) are held in four registers. If we start T at $x_0 = 1$, then after 100 bounds (and 5 minutes) he has reached $x_{100} = 89721$, and travelled a distance of $d_{100} = 1966$ (yards, say). W now takes 15 minutes to travel 6017 yards before landing in the trap. We are now relieved to find that ind $q = 1966 - 6017 = -4051$, as it should. Further experiment reveals that this trap is *perfect* and catches all kangaroos starting at or behind $x_0 = 1$; in fact, all trajectories starting at distances 0, 1, . . . , 63 yards past x_0 have merged by the time they reach $x_{15} = 15542$, a distance of only $d_{15} = 399$ yards past x_0.

To illustrate the effect of the choice of S, consider the following true Monte Carlo experiment. Let T start at the point 0, and W at an integer point chosen at random in the interval $(-L, -1)$, where L is the upper bound. The hindmost kangaroo now jumps forward by a distance chosen at random from S, and this is repeated until a collision occurs; then the number of bounds x $(\geqslant 0)$ taken by T is recorded.

According to our previous argument, x has negative exponential distribution with mean m, and standard deviation m also. Some exact values for mean m' and standard deviation σ' are:

(i) $S = (1, 2, \ldots, 2m - 1)$, $m' = m - 1$, $\sigma' = \{(m - 1)(3m - 1)/3\}^{1/2}$,

(ii) $S = (1, 2m - 1)$, $m' = (2m^2 + m - 3)/6$.

Thus (i) is satisfactory for our method (but not sparse), while (ii) is not; we give some experimental results for these and some intermediate cases (100 trials each):

(i) $S = (1, 2, \ldots, 19)$, $m = 10$, $m' = 8.68$, $\sigma' = 7.69$,

(ii) $S = (1, 19)$, $m = 10$, $m' = 37.2$, $\sigma' = 47.1$,

(iii) $S = (1, 2, 4, \ldots, 128)$, $m = 31.88$, $m' = 35.0$, $\sigma' = 27.8$,

(iv) $S = (1, 4, 16, 64)$, $m = 21.25$, $m' = 28.9$, $\sigma' = 35.1$.

From these and other experiments, we conjecture that the powers of 2 give an acceptable set S, but that sparser sets are less satisfactory.

The problem of Example 4 may seem unnatural, so we give two other situations in which the same idea might possibly be applied:

(a) to compute the order of an element x of an Abelian group, given that the order h of the group lies in the range $A < h < B$; we first seek a solution i in this range of the equation $x^i = 1$. This problem has some resemblance to the situation in [8];

(b) to factorize an integer n, given a factor bound $M > n^{1/3}$, by a version of Fermat's method [1], [5]. Let $n = pq$, where $M < p < q < n/M$, and let $(a, n) = 1$; then the equation

$$a^{2x} \equiv a^{n+1} \pmod n,$$

has a solution with $2n^{1/2} \leqslant 2x < M + n/M$, namely $x = \frac{1}{2}(p + q)$. We can find the solution (or fail to find it) in $O((n/M)^{1/2})$ steps and hence factor n (sometimes). In calling this method the Square Root Sieve we are assuming that it is possible to introduce some degree of sieving [13], [14], without losing the advantage of the square root. Certainly some improvement, at least, is possible; thus [5], if $n \equiv 11 \pmod{12}$, then $x \equiv 0$ or $6 \pmod{12}$ according as $n \equiv -1$ or $3 \pmod 8$, and we can easily make use of such a restriction on x to a single residue class.

We have not been able to experiment with this method. We remark that it likes best a number composed of two nearly equal prime factors, which the rho method [4] likes worst.

Acknowledgements. I am grateful to Professor Donald E. Knuth for some comments and references, and to the referee for his criticisms; also to the editors of *Scientific American* for some timely information about kangaroos and cryptography (August 1977).

Added in Proof. We understand that our lambda method is already known to conjurors as 'Kruskal's principle' (see *Scientific American*, February 1978). We doubt if they use a more sophisticated version, with two *herds* of kangaroos each confined to a residue class, with which we have obtained the factorization: $2^{72} - 3 = 83 \times 131 \times 294971519 \times 1472414939$.

Mathematics Department
Plessey Telecommunications Research
Taplow Court, Maidenhead
Berkshire, England

1. D. E. KNUTH, *Seminumerical Algorithms, The Art of Computer Programming*, Vol. 2, Addison-Wesley, Reading, Mass., 1969. MR 44 #3531.

2. A. RAPOPORT, "Cycle distributions in random nets," *Bull. Math. Biophys.*, v. 10, 1948, pp. 145–157.

3. B. HARRIS, "Probability distributions related to random mappings," *Ann. Math. Statist.*, v. 31, 1960, pp. 1045–1062.

4. J. M. POLLARD, "A Monte Carlo method for factorization," *BIT*, v. 15, 1975, No. 3, pp. 331–335. MR 52 #13611.

5. R. K. GUY, "How to factor a number," *Proc. Fifth Manitoba Conf. on Numerical Math.* (Winnipeg, Man., 1975), Univ. of Calgary, Dept. of Math., Statist. and Comp. Sci., Research paper no. 301, Dec. 1975.

6. J. S. DEVITT, *Aliquot Sequences*, M. Sc. Thesis, Dept. of Math. and Stat., Univ. of Calgary, May 1976.

7. D. E. KNUTH, *Sorting and Searching, The Art of Computer Programming*, Vol. 3, Addison-Wesley, Reading, Mass., 1973.

8. D. SHANKS, "Class number, a theory of factorization, and genera," *Proc. Sympos. Pure Math.*, vol. 20, Amer. Math. Soc., Providence, R. I., 1970, pp. 415–440. MR 47 #4932.

9. J. M. POLLARD, "Theorems on factorization and primality testing," *Proc. Cambridge Philos. Soc.*, v. 76, 1974, pp. 521–528. MR 50 #6992.

10. W. DIFFIE & M. E. HELLMAN, "New directions in cryptography," *IEEE Trans. Information Theory*, vol. IT-22, No. 6, Nov. 1976, pp. 644–654.

11. J. C. P. MILLER, "On factorization, with a suggested new approach," *Math. Comp.*, v. 29, 1975, pp. 155–172.

12. MICHAEL A. MORRISON & JOHN BRILLHART, "A method of factoring and the factorization of F_7," *Math. Comp.*, v. 29, 1975, pp. 183–205.

13. D. H. LEHMER, "An announcement concerning the Delay Line Sieve DLS 127," *Math. Comp.*, v. 20, 1966, pp. 645–646.

14. D. H. LEHMER & EMMA LEHMER, "A new factorization technique using quadratic forms," *Math. Comp.*, v. 28, 1974, pp. 625–635.