NOTE

Random Access to a Random Number Sequence

1. INTRODUCTION

We offer an extension of the method in a recent note by Koniges and Leith [1] for a direct jump to any kth element of a linear congruential muliplier pseudo-random number sequence [2] of form

$$\chi_{k+1} = \lambda \chi_k + c \bmod m, \qquad k \geqslant 0, \tag{1}$$

where λ (the multiplier), c (the additive constant), χ_0 (the seed), and m (the modulus) are given integers. Often, $m = 2^n$, where n is the number of bits in the mantissa of a machine word. In this note all arithmetic is assumed exact, mod m. The solution to Eq. (1),

$$\chi_k = \lambda^k \chi_0 + c \frac{\lambda^k - 1}{\lambda - 1}, \tag{2}$$

has two addends. Once contains the seed χ_0, the other has the additive constant c. We express the solution as

$$\chi_k = X_k + \Sigma_k, \tag{3}$$

where

$$X_k = \lambda^k \chi_0, \tag{4}$$

$$\Sigma_k = c(1 + \lambda + \lambda^2 + \lambda^3 + \cdots + \lambda^{k-1}) = c \frac{\lambda^k - 1}{\lambda - 1}. \tag{5}$$

The method (independently devised and used by us) involves the precalculation and storage of two arrays: one contains powers of the multiplier λ; the other has partial sums from Eq. (5). The binary digits of k specify how elements from these arrays are selected and combined to evaluate Eqs. (4) and (5). In [1] the authors note that for the common modulus, $m = 2^{48}$, and k random, one needs on average 24 multiplications to evaluate Eq. (4). For nonzero c, another 24 multiplications and 24 additions are needed for Eq. (5), although here the details were a bit sketchy. This seems a modest effort (certainly if compared to k passes through Eq. (1)!), but if large numbers of calculations are required, such as in [3], even more efficient methods may be desired.

We present a more efficient approach, a model which includes the binary method discussed above as the simplest case. We let k be represented in any convenient number base; the larger the base, the fewer the digits of k, and the less the ensuing arithmetic. For a small one-time cost of pre-calculating and storing somewhat larger arrays of needed data, the number of multiplications and additions needed to evaluate Eqs. (4) and (5) can be reduced several fold. For example, in base 8 the number of multiplications needed drops from 24 to 14. With base 256 the number of multiplications drops to 6.

2. NOTATION

In the following we assume all parameters and variables are non-negative integers less than m, that all calculations are mod m, and that routines for exact integer multiplications and additions mod m are available. Let b equal the number base in which k is expressed, and J be equal to the largest j such that b to the jth power is less than m. Then,

$$\begin{aligned} k &= d_J b^J + d_{J-1} b^{J-1} + \cdots + d_2 b^2 + d_1 b^1 + d_0 b^0 \\ &= (d_J, d_{J-1}, \ldots, d_2, d_1, d_0), \end{aligned} \tag{6}$$

where d_j is the value of the jth digit of k in base b. To clarify our argument, we will occasionally illustrate the process using the example,

$$\begin{aligned} &m = 2^{48}, \qquad b = 8, \qquad J = 15 \\ &k = 1537 \text{ (base 10)} = 3045 \text{ (base 8)} = 3045b \\ &\quad = 0 \times 8^{15} + 0 \times 8^{14} + \cdots + 3 \times 8^3 + 0 \times 8^2 + 4 \times 8^1 + 5 \times 8^0 \\ &\quad d_0 = 5, \quad d_1 = 4, \quad d_2 = 0, \quad d_3 = 3, \quad d_j = 0, \quad j \geqslant 4. \end{aligned} \tag{7}$$

3. THE EXTENDED MODEL

We wish to evaluate the pseudo-random number specified by Eqs. (4) and (5) for an arbitrary index k. We will show that the digits of k, expressed in base b, prescribe a method for evaluating Eqs. (4) and (5) with arithmetic efficiency proportional to $\log_2 b / \log_2 m$. For each nonzero

0021-9991/92 $3.00

digit, Eqs. (4) and (5) require two multiplications and one add, of elements from two matrices, P and S, containing precalculated values of powers of the multiplier and sums, respectively. The calculation of P and S is a one-time overhead, since P and S for a given generator can be permanently stored in a table.

4. THE P MATRIX

The elements of P are specified powers of the multiplier λ,

$$P_{i,j} = \lambda^{ib^j}, \qquad i = 0, ..., b-1; \qquad j = 0, ..., J. \tag{8}$$

By inspection, $P_{0,j} = 1$ for all j, and $P_{1,0} = \lambda$. The remaining elements follow from recursion relations. The columns are calculated in order, starting with $j = 0$:

$$P_{i+1,j} = P_{1,j} \times P_{i,j}, \qquad i = 1, b-2.$$

To advance j, use

$$P_{1,j+1} = P_{1,j} \times P_{b-1,j}, \qquad j = 1, J-1.$$

5. THE S MATRIX

If c is nonzero, a matrix S containing elements that are the partial sums in Eq. (5) must also be precomputed. Let

$$S_{i,j} = \frac{\lambda^{ib^j} - 1}{\lambda - 1}, \qquad i = 0, b-1; \qquad j = 0, J$$
$$= 1 + \lambda + \lambda^2 + \cdots + \lambda^{(ib^j)-1}. \tag{9}$$

By inspection, $S_{0,j} = 0$ for all j, and $S_{1,0} = 1$. The remaining elements follow from recursion relations. The columns are calculated in order, starting with $j = 0$,

$$S_{i+1,j} = S_{i,j} + P_{i,j} \times S_{1,j}, \qquad i = 1, b-2.$$

To advance j, use

$$S_{1,j+1} = S_{b-1,j} + P_{b-1,j} \times S_{1,j}, \qquad j = 1, J-1.$$

6. EVALUATING X_k

The multiplier term X_k given by Eq. (4) is evaluated by using the expression for k in base b given by Eq. (6) as the exponent to express λ^k as the product of elements of the matrix P (one factor from each column j, with row index $i = d_j$). One obtains

$$X_k = \lambda^k \chi_0 = P_{d_J,J} \times P_{d_{J-1},J-1} \times \cdots \times P_{d_0,0} \times \chi_0. \tag{10}$$

For our example, $k = 3045b$, $m = 2^{48}$ the result (all integers base 8) is

$$X_{3045} = P_{3,3} \times P_{0,2} \times P_{4,1} \times P_{5,0} \times \chi_0$$
$$= \lambda^{3000} \times \lambda^{40} \times \lambda^5 \times \chi_0$$
$$= \lambda^{3045} \times \chi_0.$$

Note that $P_{0,j} = 1$, so the factor $P_{0,2}$ and the (unwritten) factors $P_{0,j}$ for $j = 4, ..., 15$ can be skipped.

7. EVALUATING Σ_k

Use nested multiplication to express Eq. (5) in terms of the partial sums and the powers of the multiplier previously calculated and stored in $S_{i,j}$ and $P_{i,j}$, respectively. The result is

$$\Sigma_k = c[S_{d_J,J} + P_{d_J,J}[S_{d_{J-1},J-1} + P_{d_{J-1},J-1}[\cdots$$
$$[S_{d_1,1} + P_{d_1,1}[S_{d_0,0}]\cdots]]]]. \tag{11}$$

From each matrix there is one entry from each column j with row index $i = d_j$, except $P_{d_0,0}$ does not appear. The result for the example $k = 3045b$, $m = 2^{48}$, is

$$\Sigma_{3045} = c\left[\frac{\lambda^{3045} - 1}{\lambda - 1}\right] = c[1 + \lambda + \cdots + \lambda^{3045}]$$
$$= c[1 + \lambda + \cdots + \lambda^{2777} + \lambda^{3000}[1 + \lambda + \cdots + \lambda^{44}]]$$
$$= c[1 + \lambda + \cdots + \lambda^{2777} + \lambda^{3000}[0 + 1$$
$$\times [1 + \lambda + \cdots + \lambda^{37} + \lambda^{40}[1 + \lambda + \cdots + \lambda^4]]]]$$
$$= c[S_{3,3} + P_{3,3}[S_{0,2} + P_{0,2}[S_{4,1} + P_{4,1}[S_{5,0}]]]]. \tag{12}$$

Note that $S_{0,2} = 0$ and $P_{0,2} = 1$, so the $j = 2$ terms can be skipped. Similarly, the terms for $j = 4, 15$ (not explicitly shown) do not contribute.

8. SUMMARY

We have expanded the method in a recent note by Koniges and Leith [1] for a direct jump to any desired kth element χ_k of the standard linear congruential random number generator specified by Eqs. (1) and (2). The digits of k (base b) in Eq. (6) are indices for selecting elements from two precalculated arrays, P and S, Eqs. (6) and (7). These elements are combined in Eqs. (10) and (11) to give the addends of $\chi_k = X_k + \Sigma_k$.

The choice of the base b determines the number of digits in k (base b). For each nonzero digit, the evaluation of Eqs. (10) and (11) require two multiplications (mod m), and one add. The larger the choice for b, the fewer the digits in k

(base b), and the less the arithmetic needed to evaluate Eqs. (10) and (11). For modulus $m = 2^{48}$, values of $b = 2$, 8, and 256 require at most 48, 16, and 6 multiplications, respectively, to evaluate Eq. (10). The expected number of multiplications is somewhat less. As noted above, only the nonzero digits of k lead to modular arithmetic calculations (zero digits imply the addition of zero and/or multiplication by unity). For a random k, base b, the probability that a random bit is nonzero is $(b-1)/b$. Thus we find that the expected number of multiplications to evaluate Eq. (10) for $b = 2$, 8, or 256 drops to 24, 14, or 5.98, respectively.

The drawback for very large base b is the storage needed for the precalculated P and S arrays, Eqs. (8) and (9). Each has $b(J+1)$ elements, where $J+1$ is the maximum number of digits in k (base b). For $m = 2^{48}$, and $b = 2$, 8, 256, there are respectively $2 \times 48 = 96$, $8 \times 15 = 120$, and $256 \times 6 = 1536$ elements in P and in S. If b is further increased, then each array size $b(J+1)$ grows rapidly. On the other hand, the $J+1$ number of multiplications (mod m) needed to evaluate Eq. (10) decreases slowly. We suggest that for a 48-bit machine word length $b = 256$ is a good compromise between table size and achievable speedup. For a standard generator with zero additive term using base 256 reduces the overall operations count from 48 if tests, 48 register shifts, 24 multiplies, and 72 logical ands to 6 register shifts, 5.98 multiplies, and 11.98 logical ands, yielding more than a fourfold increase in speed over the base 2 case.

REFERENCES

1. A. E. Koniges and C. E. Leith, *J. Comput. Phys.* **81**, 230 (1989).
2. D. E. Knuth, *The Art of Computer Programming*, Vol. 2 (Addison–Wesley, Reading, MA, 1969).
3. J. A. Viecelli and E. H. Canfield, Jr., *J. Comput. Phys.* **95**, 29 (1990).

Received February 23, 1990; revised January 29, 1991

E. H. Canfield, Jr.
J. A. Viecelli
Lawrence Livermore National Laboratory
Livermore, California 94550