JOURNAL OF PARALLEL AND DISTRIBUTED COMPUTING 6, 477–497 (1989)

Random Number Generators for MIMD Parallel Processors*

ORA E. PERCUS AND MALVIN H. KALOS

Courant Institute of Mathematical Sciences, New York University, 251 Mercer Street, New York, New York 10012

Received June 30, 1987

We discuss and analyze issues related to the design of pseudorandom number generators (prn's) for MIMD (multiple instruction stream/multiple data stream) parallel processors, which are very well suited to Monte Carlo calculations. We are concerned with ensuring reproducibility of runs, providing very long sequences, and assuring an adequate degree of independence of the parallel streams. We consider the class of linear congruential generators

$$x_{n+1,i} \equiv a x_{n,i} + b_i \bmod m$$

and analyze the effect that different choices of b_i have on the correlation properties between such streams. We derive a spectral test ν_t for t parallel linear congruential generators, a modification of Knuth's Algorithm S. From this, we prove a good lower bound for $\nu_2^* = \min_{\text{all pairs } (i,j)} \nu_2(i, j)$ for certain choices of b_i's. The set of the largest r primes p_i, $i = 1, \ldots, r$, satisfying $p_i < \sqrt{m/2}$, where m is the period length of each generator, gives a lower bound $O(m^{1/2})$ to the correlation between a pair of corresponding elements in any two streams. An alternative choice, $b_i = d^i \bmod m$ for $d = m^{1/2} + 1$ gives a bound $O(m^{1/2}/(t-1))$ which will be satisfactory for small numbers of streams. Finally, we construct a spectral test for correlations between $x_{n,i}$ and $x_{n+k,i+l}$, but derive no analytic prescriptions from it.

1. INTRODUCTION

Designing pseudorandom number generators for current and future parallel processors presents interesting new challenges. We are interested in generators for the class of machines called MIMD (multiple instruction stream/multiple data stream) in which a number of processors are capable of essentially in-

* This work was supported by the Applied Mathematical Sciences Program of the U.S. Department of Energy under contract DE-AC02-76ER03077.

0743-7315/89 $3.00

dependent computations. Such architectures are very well suited to Monte Carlo calculations, since potentially independent realizations or "histories" may be followed on each processor. They are likely to be more suitable than the current generation of "vector" supercomputers for those Monte Carlo calculations with substantial logical complexity, i.e., those with data-dependent or random number determined branches. One example of an MIMD architecture is the NYU Ultracomputer; this is a projected architecture that will be scalable up to thousands of processors with features that make it attractive for Monte Carlo calculations of all kinds. The issues discussed in this paper are applicable to other parallel machines as well, including existing vector-parallel machines (such as the various Crays, multipipe Cyber 205s, the projected ETA 10, the IBM 3094-400, and various hypercube machines [1]). Nevertheless, the emphasis will be on problems associated with very large numbers of processors.

The new problems posed by parallel computing are the following: for asynchronous and multiprogrammed machines special care is needed to ensure reproducibility of runs; very long sequences may be necessary; finally, attention to the independence of the parallel streams used on different processors is required.

In what follows we distinguish between the computational "processes" and the physical computers of which the ensemble is made. This is necessary because in the class of architectures that we consider, there is no predictable mapping of processes to processors.

To simplify somewhat, two approaches can be considered for the generation of pseudorandom sequences: either one (or a few) processes are dedicated to generating the numbers and other processes consume their output or else each process generates its own. The first is less desirable from several points of view. It may be difficult to balance the speed of consumption and production; it may be difficult to communicate the sequences from producers to consumers. Above all, if there is any degree of asynchronism, the pattern of assignment of random numbers to processes may not be reproducible from one run to another.

It has been argued that reproducibility is not necessary for Monte Carlo calculations. After all, if the results are not predictable, what is bad about the trivial additional feature of nonreproducibility from run to identical run? Put another way, if the results are meaningful only within some statistical error bound, why be concerned whether a run can be exactly repeated? One answer, at least for that large class of problems that have the property of logical complexity mentioned above, is that the potential for exact reproducibility is extremely helpful (probably essential) for debugging. The logical errors that often occur in such programs are difficult to isolate if the run cannot be exactly retraced. An additional advantage of exact reproducibility is that it enhances confidence in the validity of a ported program. It is possible to

program these algorithms in a style that produces the same results on machines of different sizes, even on machines in which the number of processors available to a specific calculation varies unpredictably during the course of the run. It is necessary, of course, that the initial state of the computation (including the parameters of the generators) be set to be the same in alternative runs. This requirement of reproducibility puts an additional restriction on the class of random number generators that may be used. We assert that it is a desirable feature, and we shall see that it can be satisfied.

Another issue arises from the consideration of Monte Carlo calculations that simulate branching processes or in which branching processes are used as a technical device. Natural stochastic processes include the creation and destruction of particles and radiation (in cascades of high-energy particles, and in nuclear fission, to cite two significant examples.) The straightforward simulation of such phenomena leads to programs in which the one random walker may become two or more, each of which must be followed, essentially independently. In addition, effective Monte Carlo treatment of some problems leads naturally to the use of branching processes (e.g., in Green's function Monte Carlo [2]). Importance sampling of low-probability events is often accomplished in part by the use of branching. For example, in the calculation of the passage of radiation through thick media, the techniques called "splitting" and "Russian Roulette," colorful names for the birth and death of random walkers, are used to enhance the chance that some walker will survive through the medium while at the same time assuring an unbiased estimate of any property of the emerging radiation [3]. Finally, we note that Fredricksen *et al.* [4] cite several examples where calculations are more reproducible or where perturbations can be calculated more precisely when branches are inserted in the "natural" flow of the simulation.

On parallel computers, an important consideration that makes for efficient utilization of all processors is "load balancing." That is the body of techniques that guarantee that all processors carry out approximately equal work. In calculations in which branching processes appear, load balancing is likely to require that multiple random walkers that descend from some parent be simulated on processors other than the one that simulated the parent. Then, reproducibility of the calculation cannot be assured if the random number generators are associated with the physical processors. This is also true when the number of processors varies out of the program's control and the simulations migrate from processor to processor.

The necessary alternative is that random number generation be associated with the simulation processes (in the jargon of parallel computing, with the computational processes or "threads of control"), that they migrate with the simulation to other processors, and that new generators be created as often as necessary. In particular, for branching walks, a new sequence should be created for each descendant random walker that may migrate to other pro-

cessors. This was first pointed out by Zhong and Kalos [5] and leads to the concept of a pseudorandom tree, in which one generator is used for "intra-process" random numbers, another for initialization of these new streams. We suggest that it is better to think of the whole calculation to be mapped onto a tree, each node of which is associated with the initiation of a new stream. The streams are not necessarily a part of the tree. A "conventional" parallel Monte Carlo in which one stream is associated with each of several or many processes, but in which no new streams are created later, will use only the left successors of the root.

Future highly parallel MIMD will be used for very large calculations that will require very large aggregate numbers of random variables. A standard pseudorandom sequence may be adequate for each process. It will be best if each sequence can be guaranteed to have no subsequence in common with any other (or at least that overlapping sequences be very rare). Thus it will be impractical simply to use subsequences of consecutive elements of a single congruential generator unless its period is very long indeed.

If the different streams are not independent, several fresh difficulties arise. The first is easy to see by imagining the extreme case where one process uses a sequence in common with another in exactly the same way. The additional computation would produce exactly the same results and would, therefore, not reduce the variance. The "efficiency" of the parallel computation would be decreased. A similar, if more subtle, effect would be seen were there a statistical correlation among corresponding members of different streams. If either of these situations occurred, another possibly serious error would arise: the variance which is an important result of any Monte Carlo calculation would be incorrect when computed assuming that the results of separate replications are statistically independent. Finally, in calculations in which multiple descendants of a walker are treated with different pseudorandom streams, fluctuation properties of the process will be incorrectly treated if statistical correlations exist between pairs of corresponding elements of the streams. We must thus consider the issue of providing thousands (possibly many more to support load balanced branching) of sequences that are independent in these respects.

Fredrickson *et al.* [4] have proposed a specific way to construct pseudorandom trees, which results in what they call a "Lehmer tree." Congruential generators are used for both branches. Starting with x_0, the root of the tree, every element has a left branch and right branch that are calculated as

$$x_L \equiv a_L x + b_L \bmod m$$

$$x_R \equiv a_R x + b_R \bmod m.$$

The constants a_L, a_R, b_L, b_R, and m determine the tree. The right successors are used within a process. New parallel sequences are seeded from the left

successor. Thus, the constants a_L, b_L must be chosen to satisfy the criteria for a good multiplicative congruential generator. However, because all the right sequences use the same parameters except the seed, they are subsequences of a single generator. The authors proved a necessary condition for nonoverlapping of these subsequences. But the total number of entries is at most the original period. Unless a very long period is used (which will require multiprecision arithmetic on current machines) the sequences will not be long enough when very many branches of the tree are encountered. This is the case with which we are concerned here.

Alternative methods are known that give very long sequences. Tausworthe generators [6, 7] suggest themselves. Unfortunately, they are difficult to initialize so that it is not clear how to provide very many independent sequences. Furthermore, in preparing for the migration of processes, the entire list that specifies the state of the generator must be copied.

Composite generators proposed by Marsaglia [8] will also yield long sequences. The state is normally shorter than that in a typical Tausworthe generator, but again the theory for obtaining independent sequences is not available.

This paper explores a simple proposal for the generation of pseudorandom trees, namely that a new sequence to be used within a process differ from its parallel siblings simply in using a different additive constant. Our investigations began with the observation that changing the constant guarantees that the original sequence be completely reordered, so that overlapping sequences cannot occur. This is a very simple algorithm, which satisfies all the practical requirements that we have set, providing that some measure of the independence can be assured.

We explore the issue of independence and propose several pragmatic solutions. Preliminary testing (to be described elsewhere) supports the claim that they are satisfactorily independent for many purposes.

In Section 2, we derive the spectral test ν_t for t parallel linear congruential generators. ν_t is a measure of the largest wavenumbers or smallest scale of resolution, at which the points generated in the t-dimensional space can be regarded as uniform. Thus large ν_t indicates a high degree of randomness. The t linear congruential generators use the same multiplicative constant and have t different additive constants b_i, $i = 1, \ldots, t$. The new spectral test turns out to be a modification of Knuth's [9] Algorithm S. Section 3 deals with the variance reduction problem in its simplest form, namely for a function of one variable, to demonstrate the need to choose the b_i's ($i = 1, \ldots, t$), such that all $\binom{t}{2}$ pairs of a t-component pseudorandom vector satisfy the spectral test. In Section 4, we prove our basic theorem, which provides a good lower bound for $\nu_2^* = \min_{\text{all pairs } (i,j)} \nu_2(i, j)$ for certain choices of b_i's and we then proceed by suggesting concrete methods for choosing the t different b_i's. One is that of choosing the set of the largest t primes p_i, $i = 1, \ldots, t$, satisfying p_i

$< \sqrt{m/2}$, where m is the period length of each generator. Section 5 gives bounds for multiplets, for b_i's satisfying certain conditions, whereas Section 6 deals with a very simple specific choice, $b_i = d^i \bmod m$ for $d = \sqrt{m} + 1$. This power method, although very convenient computationally, becomes inferior when t is large, as seen from the comparison we have presented. Further related results are given in Section 7.

2. The Spectral Test for Parallel Generators

Here, we derive a modification of Algorithm S (see [10]) which tests the resolution at which vectors of prn's coming from similar chains, modified only by different additive constants, can be regarded as uniformly distributed. We assume throughout that we already have a "good multiplier" a, so that each sequence, when considered by itself, passes the spectral tests. This assumption will, however, not play any direct role in our discussion. Consider

$$x_{n+1,i} \equiv a x_{n,i} + b_i \bmod m, \tag{2.1}$$

$i = 1, \ldots, t$, b_i odd, $m = 2^\beta$, a a good multiplier, and assume that each sequence has maximum period (i.e., $a \bmod 4 = 1$) so that all values $0, \ldots, m-1$ occur. Then

$$x_{n,i} \equiv x_{0,i} + \gamma_i \frac{a^n - 1}{a - 1} \bmod m, \tag{2.2}$$

where

$$\gamma_i \equiv b_i + (a-1)x_{0,i}, \qquad i = 1, \ldots, t. \tag{2.3}$$

Since γ_i is relatively prime to m, γ_i^{-1} is a unique integer mod m and, therefore, we can write

$$x_{n,j} \equiv \gamma_1^{-1}\gamma_j x_{n,1} + (x_{0,j} - \gamma_1^{-1}\gamma_j x_{0,1}) \bmod m. \tag{2.4}$$

For any integers $s_1, \ldots, s_t$, then

$$\sum_{j=1}^{t} s_j x_{n,j} \equiv \gamma_1^{-1} x_{n,1} \sum_{j=1}^{t} s_j\gamma_j + \sum_{j=1}^{t} s_j(x_{0,j} - \gamma_1^{-1}\gamma_j x_{0,1}) \bmod m$$

$$\equiv \gamma_1^{-1} x_{n,1} \sum_{j=1}^{t} s_j\gamma_j + C \bmod m, \tag{2.5}$$

where $C \equiv \sum_{j=1}^{t} s_j(x_{0,j} - \gamma_1^{-1}\gamma_j x_{0,1}) \bmod m$ is a constant with respect to n.

Consider the t-dimensional discrete vector space V (over the field of integers

mod m) where each coordinate axis is of length m, i.e., $V = \{\mathbf{x}\}$, $\mathbf{x} = (x_1, \ldots, x_t)$, and x_i can assume any of the m values $0, 1, \ldots, m-1$. One can define a probability measure $g(\mathbf{x})$ for all $\mathbf{x}$ in V. If the vectors are uniformly distributed, then

$$g(\mathbf{x}) = \frac{1}{m^t}.$$

Now, for any integers $(s_1, \ldots, s_t)$mod m, define

$$f(s_1, \ldots, s_t) \equiv E\left(\exp\left(-\frac{2\pi i}{m}\sum_{j=1}^{t} s_j x_j\right)\right);$$

therefore, here

$$\begin{aligned} f(s_1, \ldots, s_t) &= \frac{1}{m^t}\sum_{\substack{\{x_j\}\\ j=1,\ldots,t}} \exp\left(-\frac{2\pi i}{m}\sum_{j=1}^{t} s_j x_j\right) \\ &= \frac{1}{m^t}\sum_{\substack{x_j=0\\ j=1,\ldots,t}}^{m-1} \exp\left(-\frac{2\pi i}{m}\sum_{j=1}^{t} s_j x_j\right) \\ &= \frac{1}{m^t}\prod_{j=1}^{t}\sum_{x=0}^{m-1} \exp\left(-\frac{2\pi i}{m} s_j x\right) \\ &= \prod_{j=1}^{t} \delta_{0, s_j \bmod m}. \end{aligned}$$

For the sequence (2.1), we have instead

$$\bar{f}(s_1, \ldots, s_t) = \sum_{\mathbf{x}\in V} \exp\left(-\frac{2\pi i}{m}\sum_{j=1}^{t} s_j x_j\right)\bar{g}(x_1, \ldots, x_t),$$

where

$$\bar{g}(x_1, \ldots, x_t) = \begin{cases} 1/m & \text{if} \quad \mathbf{x} \in \{\mathbf{x}_n, n = 1, \ldots, m\} \\ 0 & \text{otherwise} \end{cases}$$

and $\mathbf{x}_n = (x_{n,1}, \ldots, x_{n,t})$, $x_{n,i}$ defined in (2.2). Thus,

$$\bar{f}(s_1, \ldots, s_t) = \frac{1}{m}\sum_{n=1}^{m} \exp\left(-\frac{2\pi i}{m}\sum_{j=1}^{t} s_j x_{n,j}\right),$$

or using (2.5)

$$\bar{f}(s_1, \ldots, s_t) = \exp\left(-\frac{2\pi iC}{m}\right)\frac{1}{m}\sum_{n=1}^{m}\exp\left(-\frac{2\pi i}{m}\gamma_1^{-1}x_{n,1}\sum_{j=1}^{t}s_j\gamma_j\right).$$

But, the set $\{x_{n,1}\}_{n=1}^{m}$ is equal to the set of integers $\{0, \ldots, m-1\}$. Hence

$$\bar{f}(s_1, \ldots, s_t) = \exp\left(-\frac{2\pi iC}{m}\right)\sum_{x=0}^{m-1}\exp\left(-\frac{2\pi i}{m}\gamma_1^{-1}x\sum_{j=1}^{t}s_j\gamma_j\right)$$

$$= \exp\left(\frac{-2\pi iC}{m}\right)\delta_{0,\gamma_1^{-1}\sum_{j=1}^{t}s_j\gamma_j \bmod m}$$

or

$$|\bar{f}(s_1, \ldots, s_t)| = \begin{cases} 1 & \text{if} \quad \gamma_1^{-1}\sum_{j=1}^{t}s_j\gamma_j = 0 \bmod m \\ 0 & \text{otherwise.} \end{cases}$$

The modified spectral test will then be

$$\nu_t = \min_{\substack{(s_1,\ldots,s_t) \\ \ni(s_1,\ldots,s_t)\neq(0,\ldots,0)}} \{\sqrt{s_1^2 + \cdots + s_t^2}\,|\, s_1 + \gamma_1^{-1}\gamma_2 s_2 + \cdots + \gamma_1^{-1}\gamma_t s_t = 0 \bmod m\}. \quad (2.6)$$

$2\pi\nu_t$ is the magnitude of the smallest wave vector that is present in a Fourier decomposition of the point distribution on the t-dimensional unit hypercube. The distribution is therefore uniform at a resolution of $1/\nu_t$ compared to $1/m$ for the ideal distribution $g(\mathbf{x})$.

Equation (2.6) can be computed using Knuth's Algorithm S (see [10, p. 98, cf. 3.3.4]), since his algorithm does not use the fact that the coefficients are powers of a.

For a vector of t components and $m = 2^\beta$ (e.g., $\beta = 48$) the highest accuracy with which uniformity can be achieved is β/t $(48/t)$ bits. (If all 2^{48} distinct points are distributed over the t-dimensional unit hypercube, with spacing 2^{-l}, then this implies $2^{48} = (2^l)^t$ or $l = 48/t$.)

We will show later that this uniformity can be achieved not only for a two- or three-component pseudorandom vector but also simultaneously for all $\binom{t}{2}$ pairs or $\binom{t}{3}$ triplets, respectively, chosen from a t-component pseudorandom vector.

3. The Spectral Test for Minimum Variance

In a frequent type of computation, a problem is how to choose the b_i's (or γ_i's), $i = 1, \ldots, t$, such that all $\binom{t}{2}$ pairs $(x_{n,i}, x_{n,j})$ are uniformly distributed

at high levels of accuracy. This problem arises in the following fashion: Given

$$x_{1,1}, x_{2,1}, \ldots, x_{n,1}, \ldots, x_{m,1} \text{ obtained by using } b_1$$
$$\vdots$$
$$x_{1,t}, x_{2,t}, \ldots, x_{n,t}, \ldots, x_{m,t} \text{ obtained by using } b_t.$$

Let $f_{(1)}$ be a random variable assuming the values $f(x_{1,1}), \ldots, f(x_{m,1})$

$f_{(2)}$ be a random variable assuming the values $f(x_{1,2}), \ldots, f(x_{m,2})$

$\vdots$

$f_{(t)}$ be a random variable assuming the values $f(x_{1,t}), \ldots, f(x_{m,t})$.

Then

$$\operatorname{Var}\{\sum_{i=1}^{t} f_{(i)}\} = \sum_{i=1}^{t} \operatorname{Var} f_{(i)} + \sum_{\substack{i,j \\ i \neq j}} \operatorname{cov}(f_{(i)}, f_{(j)}).$$

In order for the generators to behave independently, with respect to this computation, we require

$$\operatorname{cov}(f_{(i)}, f_{(j)}) = 0 \qquad \text{for all pairs } (i, j). \tag{3.1}$$

We will show now that (3.1) is reduced to a condition on the function f that depends on the values of $\nu_2(i, j)$ for all $\binom{t}{2}$ pairs for (i, j).

We can represent $f(x)$ as

$$f(x) = \sum_{l=0}^{m-1} f_l \exp\left(\frac{2\pi i}{m} lx\right)$$

since $f(x)$ is a periodic function of period m. For simplicity, let us consider the pair (1, 2) and compute $\operatorname{cov}(f_{(1)}, f_{(2)})$,

$$\operatorname{cov}(f_{(1)}, f_{(2)}) = \frac{1}{m} \sum_j f(x_{j,1}) f(x_{j,2}) - \frac{1}{m^2} \sum_j f(x_{j,1}) f(x_{j,2})$$

$$= \frac{1}{m} \sum_{j,l,l'} f_l f_{l'} \exp\left(\frac{2\pi i}{m} (lx_{j,1} + l'x_{j,2})\right)$$

$$- \frac{1}{m^2} \sum_{\substack{l,l' \\ j,k}} f_l f_{l'} \exp\left(\frac{2\pi i}{m} (lx_{j,1} + l'x_{k,2})\right)$$

but since $x_{j,2} = \gamma_1^{-1}\gamma_2 x_{j,1} + C$ (see (2.4)),

$$\operatorname{cov}(f_{(1)}, f_{(2)}) = \frac{1}{m}\sum_{j,l,l'} f_l f_{l'} \exp\left(\frac{2\pi i}{m}(l x_{j,1} + l'\gamma_1^{-1}\gamma_2 x_{j,1} + l'C)\right)$$

$$- \frac{1}{m^2}\sum_{\substack{l,l' \\ j,k}} f_l f_{l'} \exp\left(\frac{2\pi i}{m}(l x_{j,1} + l' x_{k,2})\right)$$

$$= \frac{1}{m} m \sum_{l,l'} f_l f_{l'} \exp\left(\frac{2\pi i l' C}{m}\right)\delta_{0,(l+l'\gamma_1^{-1}\gamma_2)\bmod m} - f_0 f_0$$

or

$$\operatorname{cov}(f_{(1)}, f_{(2)}) = \sum_{\substack{l,l' \\ \ni(l,l')\neq(0,0)}} f_l f_{l'} \exp\left(\frac{2\pi i}{m} l'C\right)\delta_{0,(l+l'\gamma_1^{-1}\gamma_2)\bmod m}. \tag{3.2}$$

Define

$$\nu_2(1, 2) = \min_{\substack{l,l' \\ \ni(l,l')\neq(0,0)}} \{\sqrt{l^2 + l'^2} \mid l + \gamma_1^{-1}\gamma_2 l' = 0 \bmod m\}. \tag{3.3}$$

Consider any term on the r.h.s. of (3.2) and choose $l = \max(|l|, |l'|)$. If $l < \nu_2(1, 2)/\sqrt{2}$ then $l + l'\gamma_1^{-1}\gamma_2 \neq 0$; hence, $\delta_{0,(l+l'\gamma_1^{-1}\gamma_2)\bmod m} = 0$. But if $l \geqslant \nu_2(1, 2)/\sqrt{2}$ then $\delta_{0,(l+l'\gamma_1^{-1}\gamma_2)\bmod m}$ is not necessarily zero and we therefore need to have $f_l = 0$ to satisfy $\operatorname{cov}(f_{(1)}, f_{(2)}) = 0$ in (3.2).

We conclude, therefore, that

$$f(x) \equiv \sum_{l=-\nu_2^*/\sqrt{2}}^{\nu_2^*/\sqrt{2}} f_l \exp\left(-\frac{2\pi i}{m} l x\right) \tag{3.4}$$

will satisfy (3.1): only functions with all frequency components $< \nu_2^*/\sqrt{2}$ are guaranteed to give $\sum_{\substack{i,j \\ i\neq j}} \operatorname{cov}(f_{(i)}, f_{(j)}) = 0$. Here

$$\nu_2^* = \min_{\substack{\text{all pairs } (i,j) \\ 1\leqslant i<j\leqslant t}} \nu_2(i, j). \tag{3.5}$$

4. Basic Theorem

The above demonstrates the need to choose the t b_i's (or γ_i's) such that all $\binom{t}{2}$ pairs satisfy the spectral test, i.e., such that *all* pairs $(x_{n,i}, x_{n,j})$, $i < j$, are

uniformly distributed at the highest level of accuracy achievable. This will be accomplished if ν_2^* is of the order $m^{1/2}$.

To accomplish this let us first consider the case that $x_{0,i} = 0$ for all $i = 1, \ldots, t$. Then from (2.3), $\gamma_i = b_i$. To choose the b_i's we rely on the following:

THEOREM 1. *Let* $b_i < m$, $b_j < m$, *and* g.c.d. $(b_i, b_j) = 1$. *Then*

$$\nu_2^2(i, j) = b_i^2 + b_j^2 \qquad \text{if} \qquad b_i^2 + b_j^2 < m$$

and

$$\nu_2^2(i, j) \geqslant \frac{m^2}{b_i^2 + b_j^2} \qquad \text{if} \qquad b_i^2 + b_j^2 \geqslant m.$$

Proof.

$$\nu_2^2(i, j) = \min_{\substack{\exists(s_1, s_2) \\ (s_1, s_2) \neq (0,0)}} \{s_1^2 + s_2^2 | s_1 b_i + s_2 b_j = 0 \bmod m\}.$$

$s_1 b_i + s_2 b_j = 0 \bmod m$ is equivalent to

$$s_1 b_i + s_2 b_j = km, \qquad k = \text{integer}. \tag{4.1}$$

If $k = 0$ then since g.c.d. $(b_i, b_j) = 1$, (4.1) is satisfied by $|s_1| = \alpha |b_j|$, $|s_2| = \alpha |b_i|$ for α an integer; hence

$$\min_{\substack{\exists(s_1, s_2) \\ (s_1, s_2) \neq (0,0) \text{ and } k=0}} \sqrt{s_1^2 + s_2^2} = \sqrt{b_i^2 + b_j^2}. \tag{4.2}$$

If $k \neq 0$, $\sqrt{b_i^2 + b_j^2}\sqrt{s_1^2 + s_2^2} \geqslant |s_1 b_i + s_2 b_j| \geqslant m$ or

$$s_1^2 + s_2^2 \geqslant \frac{m^2}{b_i^2 + b_j^2}. \tag{4.3}$$

If $b_i^2 + b_j^2 < m$ then (4.2) supplies a lower bound than (4.3) and must be used. If $b_i^2 + b_i^2 \geqslant m$ the reverse holds. Q.E.D.

COROLLARY 1. *If* $b_i < \sqrt{m/2}$, $b_j < \sqrt{m/2}$, *and* g.c.d. $(b_i, b_j) = 1$, *then* $\nu_2^2(i, j) = b_i^2 + b_j^2$.

COROLLARY 2. *Let* $b_i = \alpha\sqrt{m}$, $b_j = \beta\sqrt{m}$, g.c.d. $(b_i, b_j) = 1$, *and* $\alpha^2 + \beta^2 < 1$. *Then* $\nu_2^2(i, j) = (\alpha^2 + \beta^2)m$.

Note that Theorem 1 can be easily generalized to the case g.c.d. $(b_i, b_j) = c$, in which case we obtain

$$v_2^2(i,j) = \frac{1}{c^2}(b_i^2 + b_j^2) \qquad \text{if} \qquad b_i^2 + b_j^2 < mc^2$$

$$v_2^2(i,j) \geqslant \frac{m^2}{b_i^2 + b_j^2} \qquad \text{if} \qquad b_i^2 + b_j^2 \geqslant m.$$

For the case $x_{0,i} = 0$, $i = 1, \ldots, t$, we can now take advantage of Theorem 1 by choosing the t b_i's such that

$$b_i < \sqrt{m/2}$$

and g.c.d. $(b_i, b_j) = 1$ for all pairs (i,j), $1 \leqslant i < j \leqslant t$. This can be accomplished in several ways; e.g., see the following.

Let p_1 be the largest prime such that $p_1 < \sqrt{m/2}$. Let p_j be the largest prime such that $p_j < p_{j-1}$ for $j = 2, \ldots, t$. Choose $b_i = p_i$, $i = 1, \ldots, t$. Then

$$v_2^2(i,j) = b_i^2 + b_j^2$$

and

$$v_2^* = \sqrt{p_t^2 + p_{t-1}^2}.$$

Another method is that of mixing primes of the order of $\sqrt{m/2}$ with powers of small primes, e.g., $3^{\alpha_1}, 5^{\alpha_2}, 7^{\alpha_3}, \ldots$, etc., such that $3^{\alpha_1}, 5^{\alpha_2}, 7^{\alpha_3}, \ldots$, are of the order of $\sqrt{m/2}$.

Let us note that the restriction $x_{i,0} = 0$, $i = 1, \ldots, t$ is unnecessary. If $x_{i,0} \neq 0$ for some i, then using (2.3) we get $\gamma_i = (a-1)x_{0,i} + b_i = rm + v_i$, r integer, and we will choose b_j to guarantee that $v_i < \sqrt{m/2}$ and g.c.d. $(v_i, v_j) = 1$ for all $i \neq j$.

5. Bounds for Multiplets

The problem of creating a uniform distribution for a single 2-vector can be generalized in another direction. We will devise a method of choosing b_i's such that for an extensive set of triplets

$$v_3^* = O(m^{1/3});$$

of course, as explained in [10, pp. 90, 91] having a good lower bound for v_3^* does not imply a good lower bound on v_2^*. Taking this warning into account, we now present the following result:

THEOREM 2. *Let*[1]

$$x_{0,i} = 0 \qquad \textit{for all } i. \tag{5.1}$$

Define two finite sets of primes[2]

$$\textit{Set I} = \{p_l\text{: } p_l \textit{ prime, } p_l = O(m^{2/3}), \textit{ and } p_l < m^{2/3}\}$$

$$\textit{Set II} = \{p_s\text{: } p_s \textit{ prime, } p_s = O(m^{1/3}), \textit{ and } p_s < m^{1/3}\}.$$

Then

$$\nu_2(1, 2) \geqslant O(m^{1/3}) \tag{5.2}$$

whenever b_1, b_2 *are members of either the same set or different sets;*

$$\nu_3(1, 2, 3) \geqslant O(m^{1/3}) \tag{5.3}$$

whenever $b_1 \in$ *set* I, b_2, $b_3 \in$ *set* II.

COROLLARY.

$$\nu_2^* = O(m^{1/3})$$

$$\nu_3^* = O(m^{1/3})$$

whenever one of the b*'s* $\in$ *set* I *and the other two* b*'s* $\in$ *set* II.

Proof of Theorem 2. To prove (5.2) let $b_i = \alpha_i m^{1/3}$, $\alpha_i < 1$, $i = 1, 2$. Then by Theorem 1

$$\nu_2(1, 2) = m^{1/3}\sqrt{\alpha_1^2 + \alpha_2^2}.$$

Let

$$b_1 = \alpha_1 m^{1/3}, \qquad b_2 = \alpha_3 m^{2/3}, \qquad \alpha_i < 1, \qquad i = 1, 3.$$

Then

$$\nu_2^2(1, 2) \geqslant \frac{m^2}{\alpha_1^2 m^{2/3} + \alpha_3^2 m^{4/3}} \geqslant \frac{1}{2\alpha_3^2}\, m^{2/3}$$

or $\nu_2(1, 2) \geqslant c_1 m^{1/3}$. Let $b_1 = \alpha_4 m^{2/3}$, $b_2 = \alpha_3 m^{2/3}$, $\alpha_i < 1$, $i = 3, 4$. Then

$$\nu_2^2(1, 2) \geqslant \frac{m^2}{(\alpha_3^2 + \alpha_4^2)m^{4/3}} = \frac{1}{\alpha_3^2 + \alpha_4^2}\, m^{2/3}$$

which proves (5.2).

[1] Note that this assumption is not necessary and as before it is only used to simplify the statement of the theorem and the proof.

[2] "Prime" can be replaced by every pair of elements that belong to I $\cup$ II that have no common divisor.

To prove (5.3) let

$$b_1 = \beta_1 m^{2/3}, \qquad b_2 = \beta_2 m^{1/3}, \qquad b_3 = \beta_3 m^{1/3},$$

where β_i are constants of order 1;

$$\nu_3(1, 2, 3) = \min_{\substack{s_1, s_2, s_3 \\ \ni (s_1, s_2, s_3) \neq (0,0,0)}} \{\sqrt{s_1^2 + s_2^2 + s_3^2} \mid s_1 b_1 + s_2 b_2 + s_3 b_3 = 0 \bmod m\}.$$

Since $s_1 b_1 + s_2 b_2 + s_3 b_3 = 0 \bmod m$, let $s_1 b_1 + s_2 b_2 + s_3 b_3 = km$.

Case 1. $k = 0$; then

$$s_2 b_2 + s_3 b_3 = -s_1 b_1.$$

If $s_1 \neq 0$ then $|s_2 b_2 + s_3 b_3| \geqslant b_1$ and we get

$$|s_1^2 + s_2^2 + s_3^2|^{1/2} \geqslant (s_2^2 + s_3^2)^{1/2} \geqslant \frac{b_1}{\sqrt{b_2^2 + b_3^2}} = \frac{\beta_1 m^{2/3}}{m^{1/3}\sqrt{\beta_2^2 + \beta_3^2}}$$

or $\nu_3(1, 2, 3) \geqslant c_1 m^{1/3}$, where $c_1 = \beta_1/\sqrt{\beta_2^2 + \beta_3^2}$. If $s_1 = 0$ then using Theorem 1, $\nu_3(1, 2, 3) = \sqrt{\beta_2^2 + \beta_3^2} m^{1/3}$.

Case 2. $k \neq 0$; then

$$(s_1^2 + s_2^2 + s_3^2)^{1/2}(b_1^2 + b_2^2 + b_3^2)^{1/2} \geqslant |s_1 b_1 + s_2 b_2 + s_3 b_3| \geqslant m$$

or

$$\nu_3(1, 2, 3) \geqslant \frac{m}{\sqrt{\beta_1^2 m^{4/3} + \beta_2^2 m^{2/3} + \beta_3^2 m^{2/3}}} > \frac{1}{2\beta_1 m^{1/3}}.$$

This completes the proof of the Theorem.

Theorem 2 can be generalized to higher spectral tests in the following way: Let

$$b_1 = \beta_1 m^{1/4}, \qquad b_2 = \beta_2 m^{1/4}, \qquad b_3 = \beta_3 m^{1/2}, \qquad b_4 = \beta_4 m^{3/4},$$

where g.c.d. $(b_i, b_j) = 1$ for all $i \neq j$ and β_i, $i = 1, \ldots, 4$, are constants such that $\beta_i < 1$. Then

$$\nu_2(i, j) \geqslant O(m^{1/4})$$

$$\nu_3(i, j, k) \geqslant O(m^{1/4})$$

$$\nu_4(1, 2, 3, 4) \geqslant O(m^{1/4}).$$

For the case $b_1 = \beta_1 m^{1/5}$, $b_2 = \beta_2 m^{1/5}$, $b_3 = \beta_3 m^{2/5}$, $b_4 = \beta_4 m^{3/5}$, $b_5 = \beta_5 m^{4/5}$ satisfying the conditions g.c.d. $(b_i, b_j) = 1$ for all $i \neq j$, $\beta_j < 1$ constants, $i, j = 1, \ldots, 5$ we get

$$\nu_2(i, j) \geqslant O(m^{1/5})$$

$$\nu_3(i, j, k) \geqslant O(m^{1/5})$$

$$\nu_4(i, j, k, l) \geqslant O(m^{1/5})$$

$$\nu_5(1, 2, 3, 4, 5) \geqslant O(m^{1/5}).$$

6. Method of Successive Powers

Our method of choosing primes may sometimes present certain inconveniences, such as computing and storing our necessary lists of primes and retrieving them in a reproducible way in parallel. It would be valuable, therefore, to consider a selection of γ_i's that reduces the computation of ν_t to a previously solved problem.

The method now suggested for producing a large value of ν_t where $m = 2^\beta$, with β even, is to choose

$$\gamma_i = d^{i-1} \bmod m \qquad \text{for} \qquad i = 1, 2, \ldots, t, \tag{6.1}$$

where d is odd.

Then

$$\nu_t = \min_{\substack{(s_1,\ldots,s_t) \\ \ni (s_1,\ldots,s_t) \neq (0,\ldots,0)}} \{\sqrt{s_1^2 + \cdots + s_t^2} \mid s_1 + ds_2 + \cdots + d^{t-1}s_t = 0 \bmod m\}$$

and in particular

$$\nu_2(i, j) = \min_{\substack{(s_i,s_j) \\ \ni (s_i,s_j) \neq (0,0)}} \{\sqrt{s_i^2 + s_j^2} \mid s_i d^{i-1} + s_j d^{j-1} = 0 \bmod m\}.$$

Assuming w.l.o.g. $i < j$ then

$$\nu_2(i, j) = \min_{\substack{(s_i,s_j) \\ \ni (s_i,s_j) \neq (0,0)}} \{\sqrt{s_i^2 + s_j^2} \mid s_i + s_j d^{j-i} = 0 \bmod m\}.$$

We are looking for

$$\nu_2^* = \min_{\substack{(i,j) \\ 1 \leqslant i < j \leqslant t}} \nu_2(i, j).$$

Let $d = \sqrt{m} + 1$, then for every integer p,

$$d^p = p\sqrt{m} + 1 \bmod m$$

or

$$\nu_2(i, j) = \min_{\substack{(s_i,s_j) \\ \ni(s_i,s_j)\neq(0,0)}} \{\sqrt{s_i^2 + s_j^2} \mid s_i + s_j[(j - i)\sqrt{m} + 1] = 0 \bmod m\}.$$

Using Theorem 1 we get

$$\nu_2^2(i, j) \geqslant \frac{m^2}{1 + [(j - i)\sqrt{m} + 1]^2}.$$

Hence for all pairs (i, j),

$$\nu_2^2(i, j) \geqslant \frac{m^2}{1 + [(t - 1)\sqrt{m} + 1]^2}$$

with a lower bound for ν_t^* of the order of $\sqrt{m}/(t - 1)$.

Note, however, that the components of each vector are highly correlated since for $\nu_i = (i - 1)\sqrt{m} + 1 \bmod m$ we get

$$x_{n,i+1} = x_{n,i} + e_n,$$

where e_n is a constant independent of i. If lack of correlation between sequences is important in any specific application then the prime method is obviously preferable, but we have not investigated this question in detail.

Consider next a lower bound on ν_3^*. Since

$$\nu_3^* \leqslant 2^{1/6} m^{1/3}$$

we would like to ensure a lower bound again of the order of $m^{1/3}$.

Let $d = m^{1/3} + 1$ and assume $\beta = 3l$ so that d is an integer. Then for every integer p

$$d^p = \binom{p}{2} m^{2/3} + pm^{1/3} + 1 \bmod m.$$

The condition

$$s_1 + s_2 d^{j-i} + s_3 d^{k-i} \equiv 0 \bmod m$$

$(1 \leqslant i < j < k \leqslant t)$ is replaced by

$$s_1 + s_2 + s_3 + [s_2(j-i) + s_3(k-i)]m^{1/3} + \left[s_2\binom{j-i}{2} + s_3\binom{k-i}{2}\right]m^{2/3} = lm, \quad (6.2)$$

l an integer. Hence it follows that

$$s_1 + s_2 + s_3 = \alpha m^{1/3}.$$

If $\alpha \neq 0$ then $s_1^2 + s_2^2 + s_3^2 \geqslant m^{2/3}/3$ by the Schwartz inequality. If $\alpha = 0$ this implies

$$s_2(j-i) + s_3(k-i) = \beta m^{1/3}.$$

If $\beta \neq 0$ this relation (again with the Schwartz inequality) implies

$$s_1^2 + s_2^2 + s_3^2 \geqslant s_2^2 + s_3^2 \geqslant \frac{m^{2/3}}{(j-i)^2 + (k-i)^2} \geqslant \frac{m^{2/3}}{2(t-1)^2}.$$

We are therefore left with the case $\alpha = \beta = 0$, but $\beta = 0$ implies

$$s_2 = -s_3\frac{(k-i)}{(j-i)}$$

and (6.2) implies

$$s_3\left[\binom{k-i}{2} - \frac{k-i}{j-i}\binom{j-i}{2}\right]m^{2/3} = lm. \quad (6.3)$$

But since $(s_1, s_2, s_3) \neq (0, 0, 0)$, Eq. (6.3) implies

$$|s_3| \geqslant \frac{m^{1/3}}{(k-i)[(k-i-1)/2 - (j-i-1)/2]} = \frac{2m^{1/3}}{(k-i)(k-j)}$$

$$|s_2| \geqslant \frac{2m^{1/3}}{(k-j)(j-i)}$$

or

$$s_1^2 + s_2^2 + s_3^2 \geqslant \frac{4m^{2/3}}{(t-1)^2};$$

hence

$$\nu_3^* \geqslant \sqrt{m^{2/3}/2(t-1)^2} = m^{1/3}/(t-1)\sqrt{2}$$

and

$$\nu_2^* \geqslant \nu_3^*;$$

i.e., by choosing $d = m^{1/3} + 1$ we can guarantee lower bounds of order $m^{1/3}/t$ for both ν_2^* and ν_3^*. Our previous method using primes gave lower bounds independent of t (see Table I).

7. Further Results

The problem we have discussed, constructing a spectral test for the pairs $(x_{n,i}, x_{n,j})$ to be uniformly distributed and choosing the additive constants b_i such that ν_2^* will be sufficiently large, was motivated by our special interest in minimizing the variance. Using a similar technique we can construct a test for pairs of the form $(x_{n,i}, x_{n+k,i+l})$. Using (2.2),

$$x_{n,i} \equiv x_{0,i} + \gamma_i \frac{a^n - 1}{a - 1} \bmod m$$

$$x_{n+k,i+l} \equiv x_{0,i+l} + \gamma_{i+l} \frac{a^{n+k} - 1}{a - 1} \bmod m,$$

where

$$\gamma_{i+r} = b_{i+r} + (a-1)x_{0,i+r} \qquad \text{for } r \text{ integer.}$$

Therefore

$$x_{n+k,i+l} \equiv \gamma_i^{-1}\gamma_{i+l}a^k x_{n,i} + \text{constant.}$$

Now using the method that led to (2.6) we get the two-dimensional spectral test $\nu_2^{(k)}(i, i+l)$ to be

TABLE I
Comparison

	Prime method	d methods
Lower bound for ν_2^*	$\sqrt{p_t^2 + p_{t-1}^2} \sim O(m^{1/2})$	$O(m^{1/2}/(t-1))$
Lower bound for (ν_2^*, ν_3^*)	$(O(m^{1/3}), O(m^{1/3}))$	$(O(m^{1/3}/t), O(m^{1/3}/t))$
Example: $m = 2^{48}$, $t = 6134$ processors[a]		
Lower bound for ν_2^*	1.4×10^7	2736

[a] There are 6134 primes between 9.9×10^6 and 10×10^6 (see Hardy and Wright [11]).

$$\nu_2^{(k)}(i, i+l) \equiv \min_{\substack{s_1,s_2 \\ \exists(s_1,s_2)\neq(0,0)}} \{\sqrt{s_1^2 + s_2^2} \mid s_1 + \gamma_i^{-1}\gamma_{i+l}a^k s_2 = 0 \bmod m\}$$

which as in (2.6) can be computed using Algorithm S.

A natural generalization is to consider triplets either of the form $(x_{n,i}, x_{n,j}, x_{n,k})$ or of the general form $(x_{n,i}, x_{n+k,i+l}, x_{n+k+s,i+l+r})$. As before

$$x_{n,i} \equiv x_{0,i} + \gamma_i \frac{a^n - 1}{a - 1} \bmod m$$

$$x_{n+k,i+l} \equiv x_{0,i+l} + \gamma_{i+l} \frac{a^{n+k} - 1}{a - 1} \bmod m$$

$$x_{n+k+s,i+l+r} \equiv x_{0,i+l+r} + \gamma_{i+l+r} \frac{a^{n+k+s} - 1}{a - 1} \bmod m$$

$$x_{n+k,i+l} \equiv \gamma_i^{-1}\gamma_{i+l}a^k x_{n,i} + c_1$$

$$x_{n+k+s,i+l+r} \equiv \gamma_i^{-1}\gamma_{i+l+r}a^{k+s}x_{n,i} + c_2.$$

c_1 and c_2 are constants, and the spectral test $\nu_3^{(k,s)}(i, i+l, i+l+r)$ will be of the form $\nu_3^{(k,s)}(i, i+l, i+l+r) = \min_{\substack{(s_1,s_2,s_3) \\ \exists(s_1,s_2,s_3)\neq(0,0,0)}} \{\sqrt{s_1^2 + s_2^2 + s_3^2} \mid s_1 + \gamma_i^{-1}\gamma_{i+l}a^k s_2 + \gamma_i^{-1}\gamma_{i+l+r}a^{k+s}s_3 \equiv 0 \bmod m\}$.

8. Conclusions

It is gratifying to have methods that produce good lower bounds on ν_2 for all pairs in a large ensemble of parallel streams. The use of prime b_i yields a bound $\nu_2^* = O(m^{1/2})$ which is satisfactory for many calculations. The very convenient method of finding b_i from $\gamma_i = d^{i-1}$ yields good bounds $O(m^{1/2}/(t-1))$ for small numbers of streams, say up to 100. If the number of streams can be set in advance (e.g., one for each of a fixed number of processes), then each can be assigned its unique prime or its unique γ_i.

For the case discussed in the Introduction where an indefinite number of streams may be invoked, as in the general random walk with branching, where the computational load may be shared unpredictably and where reproducibility of results is required, a technical problem remains; How are primes or powers of d assigned to processes? We do not understand how to do that and ensure that the same b is not used twice. But if one accepts using b's twice with very small probability, the quality of the calculation remains high. Several schemes are possible. When one initiates a new stream, one can "hash" the x and b from the parent stream to an integer which points into a

large table of primes. That is, the state of the parent is used to give any entry in the table with equal chance. If one used either b or x alone, then the new streams would have a simple relation to the old. For example, if we use b alone, then all streams whose position on the tree of streams involves a fixed number of left steps will have the same b. If one uses x alone, then every stream started with a particular x (admittedly rare) will be exactly the same stream. Picking γ (instead of b) also serves to mix x and b.

We should also point out that we have been using, with seeming success, a completely *ad hoc* scheme in this style for generating new streams. It is

$$b_{j+1} = (a_b b_j + c_b \bmod m) \oplus x \text{ or } 1,$$

where

$$a_b = 44\ 485\ 709\ 377\ 909$$

$$c_b = 163\ 287\ 475\ 723\ 473.$$

The exclusive or $\oplus$ operation is used simply to mix the x part of the state and so avoid the predictable use of identical b's as discussed above. We have tested this generator in several ways. The most notable is in a simple soluble branching random walk. As discussed in the Introduction, descendant walkers were assigned new random streams. Eight moments of the distribution of walkers were calculated correctly, a pragmatic and stringent test of the independence of different streams.

Nevertheless, we regard our analytic results as important, if only as a beginning for a theory of parallel generators. We hope, in the future, to provide some bounds on correlations between $x_{i,j}$ and $x_{i+m,j+n}$ and in so doing to put our proposals on a still more satisfactory basis. It would also be good to have similar theory and bounds for parallel versions of other generators such as Tausworthe, lagged Fibonacci, and composite generators.

Finally, we note that these methods can also be used to improve the quality of purely sequential generators. It is well known (or ought to be; cf. [13]) that linear congruential generators that use powers of two as a modulus have strong sequential correlations between any entry and a successor delayed by a (moderately large) power of two. In our testing, we have verified that such a correlation is serious for delays of 1024 and larger. There are cases of errors in Monte Carlo calculations in which exactly such numbers of variables are used repeatedly in a fixed pattern. A simple cure is to change the additive constant randomly at intervals short enough to remove an undesirable correlation.

9. ACKNOWLEDGMENTS

We thank J. K. Percus whose many contributions and constant criticism were invaluable, A. D. Sokal for participating in many stimulating discussions on these problems, and A. Lapidus

and J. Chiabaut for interesting conversations and numerical studies. We also acknowledge the helpful remarks of anonymous referees.

REFERENCES

1. Dongara, J. J., Ed. *Experimental Parallel Computing Architectures.* North-Holland, Amsterdam, in press.
2. Ceperley, D. M., and Kalos, M. H. Quantum many-body problems. In Binder, K., (Ed.). *Monte Carlo Methods in Statistical Physics,* Vol. 1. Springer-Verlag, Berlin, 1979.
3. Kalos, M. H., and Whitlock, P. A. *Monte Carlo Methods,* Vol. I, *Basics.* Wiley, New York, 1986.
4. Fredrickson, P., Hiromoto, R., Jordon, T. L., Smith, B., and Warnock, T. Pseudorandom trees in Monte Carlo, *Parallel Computing,* Vol. 1. 1984, p. 175.
5. Zhong, Y.-Q., and Kalos, M. H. Monte Carlo transport calculations on an ultracomputer. Ultracomputer Note #46, CIMS, New York University, 251 Mercer Street, New York, NY 10012.
6. Tausworthe, R. C. Random numbers generated by linear recurrence modulo two. *Math. Comp.* **19**(1965), 201.
7. Niederreiter, H. Statistical tests for Tausworthe pseudo-random numbers. In Grossman, W., *et al.* (Eds.). *Probability and Statistical Inference.* Reidel, Dordrecht, 1982.
8. Marsaglia, G. A current view of random number generators, keynote address. *Computer Science and Statistics,* XVI, *Symposium on the Interface,* Atlanta, 1984.
9. Knuth, D. E. *The Art of Computer Programming,* Vol. 2, *Seminumerical Algorithms.* Addison–Wesley, Reading, MA, 1969.
10. Knuth, D. E. *The Art of Computer Programming,* Vol. 2, *Seminumerical Algorithms,* 2nd ed. Addison–Wesley, Reading, MA, 1981.
11. Hardy, G. H., and Wright, E. M. *The Theory of Numbers,* 4th ed. Oxford Univ. Press (Clarendon), Oxford, 1960.
12. Riesel, H. *Prime Numbers and Computer Methods for Factorization.* Birkhauser, 1985, p. 45.
13. Percus, O. E., and Percus, J. K. Long range correlation in linear congruential generators. *J. Comput. Phys.,* in press.