IEEE COMPUTER
SOCIETY REPRINT

LINEAR CONGRUENTIAL GENERATORS DO NOT PRODUCE RANDOM SEQUENCES

A.M. Frieze
R. Kannan
J.C. Lagarias

Reprinted from IEEE 1984 FOCS — PROCEEDINGS OF THE 25TH
ANNUAL SYMPOSIUM ON FOUNDATIONS OF COMPUTOR SCIENCE

IEEE COMPUTER SOCIETY
1109 Spring Street, Suite 300
Silver Spring, MD 20910

IEEE
COMPUTER
SOCIETY
PRESS

THE INSTITUTE OF ELECTRICAL AND ELECTRONICS ENGINEERS, INC

LINEAR CONGRUENTIAL GENERATORS DO NOT PRODUCE RANDOM SEQUENCES

A.M. Frieze[*], R. Kannan[**] and J.C. Lagarias[***]

[*]GSIA, Carnegie-Mellon University and Queen Mary College, London,
[**]Computer Science Department, Carnegie-Mellon University,
[***]AT&T Bell Laboratories, Murray Hill.

Abstract

One of the most popular and fast methods of generating "random" sequence are linear congruential generators. This paper discusses the predictability of the sequence given only a constant proportion α of the leading bits of the first few numbers generated. We show that the rest of the sequence is predictable in polynomial time, almost always, provided $\alpha > 2/5$.

One of the most popular and fast methods of generating "random" sequences are linear congruential generators. These work as follows: a modulas M, a multiplier a relatively prime to M and an increment c are picked. Then starting at a random "seed" X_1 one generates the sequence $\{X_i\}$ given by

$$X_{i+1} = a \cdot X_i + c (\mathrm{mod}\ M) \qquad (0)$$

(Thus the X_i are all integers between 0 and M - 1.) Knuth (Vol. 2) contains an elaborate discussion of linear congruential generators (LCG). The sequences produced by LCG's have been shown to satisfy various statistical tests of randomness for proper choices of the modulas and multiplier. (Knuth-Vol. 2). However it does not immediately follow from these that these sequences are "unpredictable" - which one would intuitively expect a random sequence to be. This aspect of randomness has been formalized by cryptographers Shamir (1980), Blum and Micali (1982), Yao (1982) and Goldreich, Goldwasser and Micali (1984). Also, the thesis that problems that can be done in random polynomial time are essentially tractable is based on the hypothesis that a deterministic polynomial time bounded process can produce sequences that are indistinguishable from truly random sequences in deterministic polynomial time. (See Cook (1983) for a discussion of this thesis). Indeed the general observation so far seems to be that probabilistic (coin-tossing) algorithms work well in practice. In view of this it is important to analyse one of the most popular random number generators - the linear congruential generator for predictability.

It has been suggested (Knuth 1980) that a way of producing secure sequences from an LCG is to output the leading part of each of the X_i's - say the leading half of the bits.† The main result of this paper is to show that this sequence is not secure. Knuth (1980), Plumstead (1982) and Reeds (1977) have considered the question of whether bits generated by linear congruential generators are predictable. Plumstead (1982) uses a clever idea to show that if all the bits of several consecutive X_i's are known, then the multiplier a can be inferred and with greater difficulty the modulas too, thus demonstrating that when all bits of X_i are announced, the sequence becomes predictable even if the modulas and multiplier are unknown. Knuth (1980) considers the problem when

†Note that if the modulas is known it is certainly insecure to output X_i, X_{i+1}, X_{i+2} for any i, for then a is given by $(X_{i+1}-X_i)^{-1} \cdot (X_{i+2}-X_{i+1})$ and thence c can be found. Here the inverse is modulo M, if the inverse does not exist, a simple modification of the expression suffices to find a.

0272-5428/84/0000/0480$01.00 © 1984 IEEE

the multiplier and modulas are unknown and only a small fraction of the bits of several consecutive X_i's are announced. For this case, he devises an exponential time algorithm to infer the hidden information. Reeds (1977) considers some special cases with fixed multipliers. Plumstead (1982) also treats the case when the trailing O(log(n)) bits of several consecutive X_i's are unknown.

To describe our result we first introduce some notation: let $n = 2m$ be the number of bits in M. We break X_i into two equal parts:

$$X_i = 2^m \cdot y_i + z_i \qquad (1)$$

where $0 \leq y_i, z_i \leq 2^m$. The problem we consider is: given M, a, c, y_1, y_2, $y_3, \ldots y_\ell$ for some ℓ, can one determine z_1 (and then of course all the X_i can be easily computed.) The main result is an algorithm A with the following properties:

1) A is deterministic polynomial time bounded. Indeed A runs in time $O(n^2 \text{lognloglogn})$.

2) It takes as input integers M, a and integers y_1, y_2 and y_3, $0 \leq y_1, y_2, y_3 \leq 2^m$ and returns an integer z_1 between 0 and 2^m or returns the answer "cannot solve the instance". (See (3) below)

3) For each M, there is a set S_M containing at least $(1 - O(M^{-(1/5)})$ of the integers modulo M such that

a) for any a in S_M, and any c, given y_1, y_2, y_3 integers in $[0, \sqrt{M}]$, there is a unique z_1, z_2 and z_3 in $[0, \sqrt{M}]$ such that x_1, x_2 and x_3 defined by (1) satisfy (0).

b) there is a polynomial-time algorithm that given a,M tests whether a is in S_M.

c) whenever $a \in S_M$, the algorithm A gives the correct (unique) answer; if $a \notin S_M$, A returns "cannot solve".

Algorithm A

We use the algorithm of Kannan (1983) to find an integer solution to

$$\begin{aligned} az_1 - z_2 + Mp_1 &= Y_1 \\ az_2 - z_3 + Mp_2 &= Y_2 \\ 0 \leq z_i \leq 2^m & \qquad i = 1, 2, 3. \end{aligned} \qquad (2)$$

where $Y_i = 2^m(y_{i+1} - ay_i - c) \pmod M$ for $i = 1,2$ and p_1, p_2 are new integer variables. (We remark that Lenstra's (1979) algorithm could take $\Omega(n^9)$ time).

Now clearly if z_1, z_2, z_3 are the "hidden bits" of an LCG then they will form a solution to (2) with suitable values for p_1, p_2. The key issue is whether or not there are any other solutions. If there are none then our method is valid.

We define the set S_M for which we know that the solution is unique.

Suppose that there is another solution (z_1', z_2', z_3', p_1', p_2') to (2). Then putting $u_i = z_i - z_i'$ for i = 1, 2, 3 we have

$$\begin{aligned} u_2 &= au_1 \quad (\text{Mod } M) \\ u_3 &= au_2 \quad (\text{Mod } M) \\ |u_i| &\leq 2^{m+1} \qquad i = 1,2,3 \end{aligned} \qquad (3)$$

where we define (Mod M), as opposed to (mod M) to be the least absolute value residue i.e. $-M/2 \leq y(\text{Mod } M) < M/2$. We can assume without loss of generality that $u_1 > 0$ (clearly if $u_1 < 0$ we replace u_1 by $-u_1$. If $u_1 = 0$ we find that $u_2 = u_3 = 0$ as $|u_i| < M$. But then $p_i = p_i'$ for $i = 1,2$ follows easily and our solutions are not distinct.)

Thus if

$$B_M = \{0 \leq a \leq M-1 : \exists x, 0 < x \leq 2^{m+1} \text{ such that } |a^i x \,(\text{Mod } M)| \leq 2^{m+1},\ i=1,2\}$$

and

$$S_M = \{0, 1, \ldots M-1\} - B_M$$

then we have

If $a \in S_M$ *then there is at most one solution to (2) and our algorithm finds it.* (4)

Our next task is to bound the size of B_M. For $0 < x \leq L = 2^{m+1}$ let

$B(x) = \{0 \leq a \leq M-1: |ax(\text{Mod } M)|, |a^2x(\text{Mod } M)| \leq L\}$.

Now

$$|B_M| \leq \sum_{x=1}^{L} |B(x)| \qquad (5)$$

as each $a \in B_M$ is counted at least once in the sum on the right hand side of (5).

Consider now a fixed x, $0 < x \leq L$ and assume first that x and M are relatively prime. Let $w = x^{-1}$ (Mod M).

Then putting $y = ax$ (Mod M) and using $a^2x = wy^2$ (Mod M) we obtain

$$|B(x)| = |X_w| \qquad (6)$$

where $X_w = \{-L \leq y \leq L: |wy^2 \text{ (Mod } M)| \leq L\}$.

We now obtain a bound for the size of $|X_w|$ which will be used with (5) and (6) to bound $|B_M|$.

Consider the function ϕ: $X_w^2 \dashrightarrow Z$ defined by

$$\phi(y_1, y_2) = w(y_1^2 + y_2^2) \text{ (Mod M)}. \qquad (7)$$

Note that

$$|\phi(y_1, y_2)| \leq 2L \qquad \text{for } y_1, y_2 \in X_w. \qquad (8)$$

Let now $\varepsilon > 0$ be an arbitrarily small positive real number. We show that there exists a_ε such that if $|u| \leq 2L$ then

$$|\phi^{-1}(u)| \leq \underbrace{a_\varepsilon L^{2+2\varepsilon}/M} \qquad (9)$$

$$\Downarrow$$

$$|X_w|^2 \leq 2L *$$

To see this consider a fixed $(y_1,y_2) \in \phi^{-1}(u)$ having the smallest value of $y_1^2 + y_2^2$. Then $(y_1', y_2') \in \phi^{-1}(u)$

if and only if

$$w(y_1'^2 + y_2'^2) = w(y_1^2 + y_2^2) \quad \text{(Mod M)}$$

if and only if

$$y_1'^2 + y_2'^2 = y_1^2 + y_2^2 \quad \text{(Mod M)}$$

if and only if

$$y_1'^2 + y_2'^2 = y_1^2 + y_2^2 + pM$$

for some integer p, $0 \leq p \leq \bar{p} = \lfloor (2L^2 - y_1^2 - y_2^2)/M \rfloor$.

Now for non-negative integer n, let $\psi(n)$ denote the number of distinct integer solutions (x,y) to the equation

$$x^2 + y^2 = n.$$

It follows that

$$|\phi^{-1}(u)| \leq \sum_{p=0}^{\bar{p}} \psi(y_1^2 + y_2^2 + pM) \qquad (10)$$

Now it is known (Le Veque (1956) for example) that for any $\varepsilon > 0$ there exists b_ε such that $\psi(n) \leq b_\varepsilon n^\varepsilon$. It follows from (10) that (9) holds with $a_\varepsilon = 2^{1+\varepsilon} b_\varepsilon$.

It then follows from (8) and (9) that

$$|X_w| \leq (4a_\varepsilon L^{3+2\varepsilon}/M)^{1/2} \qquad (11)$$

which completes the case for x and M relatively prime.

If $d=d(x) = \gcd(x,M)>1$ we find that

$$|B(x)| \overset{*}{=} d(x) \mid \{-\hat{L} \leq y \leq \hat{L} : |\hat{\omega} y^2 \text{ (Mod } \hat{M})| \leq \hat{L}\} \mid$$

where $\hat{M} = M/d$, $\hat{L} = |L/d|$ and $\hat{\omega} = (x/d)^{-1}$ (Mod $\hat{M}$).

$$a \in B(x) \Rightarrow a + i\hat{M} \in B(x), \quad i = 0, 1, \cdots d-1$$

It follows from (11) that $|B(x)| \leq d(x) (4a_{\epsilon}\hat{L}^{3+2\epsilon}/\hat{M})^{1/2}$ and hence that

$$|B_M| \leq \sum_{x=1}^{L} d(x)^{-\epsilon}(4a_{\epsilon}L^{3+2\epsilon}/M)^{1/2} \qquad (12)$$

$$\leq c_{\epsilon}(L^{5+2\epsilon}/M)^{1/2}$$

where $c_{\epsilon} = 2a_{\epsilon}^{1/2}$.

Substituting $L = 2^{m+1}$ and putting $\epsilon = 1/20$ yields $|B_M| = O(M^{4/5})$ as stated.

We note that if we are given slightly fewer than n/2 bits i.e. $|\alpha n|$ bits where $\alpha > 2/5$ then simply putting $m = |(1-\alpha)n|$ in the above analysis shows that our method works except on a set of a's of size $O(M^{2-5\alpha/2+\epsilon})$ for any $\epsilon > 0$.

We now consider the problem of testing for $a \in B_M$. This is again an integer program in a fixed number of variables. Thus $a \in B_M$ if and only if there is a solution to

$$1 \leq x \leq L$$
$$-L \leq ax + p_1M \leq L$$
$$-L \leq a^2x + p_2M \leq L$$
$$x, p_1, p_2 \text{ integer.}$$

Extensions the problem naturally arises: what if instead of half the bits we are only given a much smaller fraction of them? Then, of course we may require portions of more than 3 of the X_i's, but will a fixed number depending only on α do? We show that the answer is affirmative provided the following number theory conjecture is true:

Corresponding to any fraction $\alpha \in (0,1)$ there exists a natural number ℓ and a fraction $\delta \in (0,1)$ such that the cardinality of the set $B_{\alpha,M}$ defined below is $O(M^{\delta})$.

$$B_{\alpha,M} = \{a : 0 \leq a \leq M-1; \exists x, 0 < x \leq M^{\alpha} \text{ such that } |a^i x (\text{Mod } M)| \leq M^{\alpha},\ i = 1,2,\ldots \ell\}.$$

We have proved the conjecture when M is square free. However the conjecture is open for general M.

We next consider the case where the constant c in (0) is not known. As it turns out, we can proceed in a similar manner to the above. This time we need the first 3 numbers generated. Using the decomposition (1) we will be looking for an integer solution to

$$az_1 - z_2 + c + Mp_1 = Y_1 \qquad (13)$$
$$az_2 - z_3 + c + Mp_2 = Y_2$$
$$az_3 - z_4 + c + Mp_3 = Y_3$$

$$-M < c < M$$
$$-2^m \leq z_i \leq 2^m \qquad i = 1,2,3,4$$

where $Y_i = 2^m(y_{i+1} - ay_i) \pmod M$ for $i = 1,2,3$. We show next that if we change the definition of B_M slightly by replacing 2^{m+1} by 2^{m+2} then

$$a \in S_M \text{ implies (13) has a unique solution} \qquad (14)$$

Suppose $(z_1', z_2', z_3', z_4', c', p_1', p_2', p_3')$ is an alternative solution. Put $v_i = z_1 - z_i'$ for $i = 1,2,3,4$ and the $u_i = v_i - v_{i+1}$ for $i = 1,2,3$. It follows that (3) holds with 2^{m+1} replaced by 2^{m+2}.

Finally the case when a and possibly M are also unknown in addition to a fraction of the bits of X_i, remains an interesting open problem.

The ideas used in this paper will yield an algorithm for the case when M is odd and the trailing half of the bits are given to us. (When M is even these bits do not form a random sequence - this can be seen from basic considerations.) The sets B_M, S_M do not change.

Acknowledgment We thank Rick Statman for useful discussions and M. Blum and J. Plumstead for bringing the problem to our attention.

References

M. Blum and S. Micali, "How to generate cryptographically strong sequence of pseudo random bits?" Proceedings of the 23rd IEEE Symposium of the Foundations of Computer Science (1982).

S. Cook, "An overview of computational complexity" - 1982 ACM Turing Award lecture, Communications of the ACM Vol. 26, No. 6 June (1983) pp. 400-408.

O. Goldreich, S. Goldwasser and S. Micali, "How to construct random functions".

R. Kannan, "Improved algorithms for integer programming and related lattice problems" 15th Annual ACM symposium on theory of computing (1983) pp. 193-206.

D. E. Knuth, "Seminumerical algorithms. The art of computer programming" Vol. 2, Addison-Wesley (1969).

D. E. Knuth, "Deciphering a linear congruential encryption" Technical Report no. 024800, Stanford University (1980).

H. W. Lenstra, "Integer programming with a fixed number of variables" First announcement (1979) To appear in Mathematics of Operations research.

W. J. LeVegne, "Topics in number theory," Addison-Wesley, Mass. (1956).

J. Plumstead, "Inferring a sequence generated by a linear congruence" 23rd IEEE Symposium on the Foundations of Computer Science (1982), pp. 153-159.

J. Reeds, "Cracking a random number generator" Cryptologia, Vol. 1, Jan (1977).

A. Shamir, "On the generation of cryptographically strong pseudo random sequences" Seventh International Colloquium on Automate, Languages and Programming, (1980).

A. Yao, "Theory and applications of trapdoor functions" Proceedings of the 23rd IEEE Syposium of the Foundations of Computer Science (1982), pp. 80-91.