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 \(a\) 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 \(a > 2/5\).

One of the most popular and fast methods of generating "random" sequences are linear congruential generators. These work as follows:

a modulus \(M\), 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

\[X_{i+1} = a \cdot X_i + c \text{ (mod } M)\]  

(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 modulus 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 analyze 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 modulus too, thus demonstrating that when all bits of \(X_i\) are announced, the sequence becomes predictable even if the modulus and multiplier are unknown. Knuth (1980) considers the problem when

**Note that if the modulus is known it is certainly insecure to output \(X_{i+1}X_{i+2}\) for any \(i\), for then \(a\) is given by \((X_{i+1}X_{i+2})^{-1}\cdot (X_{i+2}X_{i+1})\) 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 \(a\).
the multiplier and modulus 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:

\[
X_i = 2^m \cdot y_i + z_i
\]

where \( 0 \leq y_i, z_i \leq 2^m \). The problem we consider is: given \( M, \alpha, \gamma_1, \gamma_2 \), for some \( \gamma \), 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 \( O(n^{2+10\log\log n}) \).

2) It takes as input integers \( M, \alpha, \gamma_1, \gamma_2 \) and \( \gamma \), \( 0 \leq \gamma_1, \gamma_2, \gamma \leq 2^m \) 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 \( \gamma \), given \( \gamma_1, \gamma_2, \gamma_3 \) integers in \( [0, M) \), there is a unique \( z_1, z_2 \) and \( z_3 \) in \( [0, M) \) 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

\[
\begin{align*}
az_1 - z_2 + MP_1 &= Y_1 \\
zs_2 - z_3 + MP_2 &= Y_2 \\
0 \leq z_i &\leq 2^m \\
\end{align*}
\]

where \( Y_i = 2^m(y_i + a\gamma_i - c) \pmod{M} \) for \( i = 1, 2 \) and \( P_1, P_2 \) are new integer variables. (We remark that Lenstra's (1973) algorithm could take \( \Omega(n^3) \) 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

\[
\begin{align*}
u_2 &= au_1 \pmod{M} \\
u_3 &= au_2 \pmod{M} \\
u_1^i &\leq 2^{m+1} \\
\end{align*}
\]

where we define \((\pmod{M})\), as opposed to \((\pmod{M})\) to be the least absolute value residue i.e. \(-M/2 \leq y(\pmod{M}) < M/2\). 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 \( ju_1 < M \). But then \( P_1 = P_1' \) for \( i = 1, 2 \) follows easily and our solutions are not distinct.) Thus if

\[
R_M = \{0 \leq a \leq M-1: 3x, 0 < x \leq 2^{m+1} \text{ such that } |a^i x (\pmod{M})| \leq 2^{m+1}, i = 1, 2\}
\]

and

\[
S_M = \{0, 1, ..., M-1\} - R_M
\]

then we have
If \( a \in \mathbb{Z}_M \) then there is at most one solution to (2) and our algorithm finds it. (4)

Our next task is to bound the size of \( B_{BM} \).

For \( 0 < x \leq L = 2^M + 1 \) let

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

Now

\[
|B(x)| \leq \sum_{x} |B(x)| \quad (5)
\]
as each \( a \in B_x \) 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 \( w = x^{-1} (\text{Mod M}) \).

Then putting \( y = ax (\text{Mod M}) \) and using \( a^2 x = wy^2 \) (Mod M) we obtain

\[
|B(x)| \leq |X_\omega| \quad (6)
\]

where \( X_\omega = \{ 0 < \omega \leq l : \text{wy}^2 \text{(Mod M)} \leq \lambda \} \).

We now obtain a bound for the size of \( |X_\omega| \) which will be used with (5) and (6) to bound \( |B_{BM}| \).

Consider the function \( \phi : X_\omega \rightarrow \mathbb{Z} \) defined by

\[
\phi(y_1, y_2) = w(y_1^2 + y_2^2) \text{ (Mod M)}. \quad (7)
\]

Note that

\[
|\phi(y_1, y_2)| \leq 2L \quad \text{for} \quad y_1, y_2 \in X_\omega. \quad (8)
\]

Let now \( e > 0 \) be an arbitrarily small positive real number. We show that there exists \( a_e \) such that if \( |u| \leq 2L \) then

\[
|\phi^{-1}(u)| \leq a_e L^{-2e/M} \quad (9)
\]

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

\[
w(y_1^2 + y_2^2) = w(y_1^2 + y_2^2) \text{ (Mod M)}
\]

if and only if

\[
y_1^2 + y_2^2 = y_1^2 + y_2^2 \text{ (Mod M)}
\]

if and only if

\[
y_1^2 + y_2^2 = y_1^2 + y_2^2 + pM
\]

for some integer \( p, 0 \leq p \leq \lambda = ((2L^2 - \lambda^2)/\lambda M) \).

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

\[y^2 + y^2 = n.
\]

It follows that

\[
|\phi^{-1}(u)| \leq \sum_{p=0}^{\lambda} |\phi(y_1^2 + y_2^2 + pM)| \quad (10)
\]

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

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

\[
|X_\omega| \leq (2L)^{1/2} \quad (11)
\]

which completes the case for \( x \) and \( M \) relatively prime.

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

\[
|B(x)| = d(x) \left\lfloor -\frac{1}{2} \sum_{\phi \in \mathbb{Z}^2 \text{ (Mod M)} \leq \lambda} 1 \right\rfloor
\]

where \( M = M/d, L = |L/d| \) and \( \phi = (x/d)^{-1} \text{ (Mod M)} \).
It follows from (11) that $|B(x)| \leq d(x)$ and hence that

$$I \leq d(x)^{e^{-2}(4aL^{3+2c/M})^{1/2}}$$

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

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

We now consider the problem of testing for $\alpha \in S_M$. This is again an integer program in a fixed number of variables. Thus $\alpha \in B_M$ 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 $\alpha$ 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 $\epsilon \in (0,1)$ such that the cardinality of the set $B_{\alpha}M$ defined below is $O(M^\theta)$.

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

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 (10) 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

$$ax_1 - x_2 + c + Mp_1 = Y_1$$
$$ax_2 - x_3 + c + Mp_2 = Y_2$$
$$ax_3 - x_4 + c + Mp_3 = Y_3$$

where $Y_i = \frac{2^m}{y_{i+1} - ay_i} \mod M$ 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

$\alpha \in S_M$ implies (13) has a unique solution (14).

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

Finally the case when $\alpha$ and possibly $M$ are also unknown in addition to a fraction of the bits of $X_4$, 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


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


