Sandia Corporation

REPRINT

GEOMETRIC PROGRESSIONS MODULO n AS RANDOM NUMBER GENERATORS

by

Donald R. Morrison

June 1958

DISCLAIMER

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency Thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

DISCLAIMER

Portions of this document may be illegible in electronic image products. Images are produced from the best available original document.

Presented at organizational meeting of the
Southwestern Section,
ASSOCIATION FOR COMPUTING MACHINERY
October 1957

SCR-22

SANDIA CORPORATION REPRINT

GEOMETRIC PROGRESSIONS MODULO n AS RANDOM NUMBER GENERATORS

by

Donald R. Morrison

June 1958

Work Performed Under AEC Contract AT-(29-1)-789

ABSTRACT

A geometric progression modulo n is defined, and methods are given for predicting its period. Its usefulness as a random number generator is discussed.

GEOMETRIC PROGRESSIONS MODULO n AS RANDOM NUMBER GENERATORS

The first decision to be made in programming a so-called random number generator is the decision as to what is required of the generator. The requirement that it generate random numbers is too severe from two points of view; it is not necessary, and it cannot be met by a programmed generator. A more suitable requirement is that the numbers generated be free of certain kinds of regularity. For most problems requiring so-called random numbers, it is most important that regularity of the most significant digits be avoided; whereas, regularity of the least significant digits is more tolerable. There are, undoubtedly, problems in which the reverse is true.

The second question to be answered is: of the generators we know how to program, which ones most consistently avoid the objectionable kinds of regularity? It is at this point that the principal advantage of the geometric progression modulo n becomes apparent, for the regularities and nonregularities in this kind of progression are known or are easily computed theoretically. This is in marked contrast with the situation with other generators for which the known properties are essentially those determined by experiment.

A geometric progression modulo n is defined in terms of three positive integers,

- a the starter,
- r the multiplier,
- n the modulus,

and the progression is written symbolically, (a, r, n) . The i th number generated is written t_i , and t_i is obtained by multiplying t_{i-1} by r , dividing by n and saving the remainder as t_i . For example, the progression $(1, 6, 100)$ is

1, 6, 36, 16, 96, 76, 56)

There are several operations which can be performed on progressions to form other progressions. Two of them are:

1. Addition of progressions with the same multiplier and modulus:

$$(a, r, n) + (b, r, n) = (a+b_{\text{mod } n}, r, n)$$

2. Scalar multiplication:

$$c (a, r, n) = (ca, r, cn)$$

The stem of a progression is the set of numbers which are generated exactly once; the loop is the set of numbers which are eventually repeated. The progression can be represented schematically as follows: The progression written above will illustrate.

The first step in the analysis of a progression is the identification of the stem and the loop. This is done by finding the greatest common divisor g_i of t_i and n . So long as t_i is in the stem, g_i will increase, but when t_i is in the loop, g_i will become constant.

Illustration (1, 6, 100)

t_i	1,	6,	36,	16,	96,	76,	56
g_i	1	2	4	4	4	4	4

In the remaining analysis it is assumed that the stem has been identified and removed, and that the progression to be analyzed is all loop. The next step is to remove any common factor of the terms and the modulus. Since the progression is all loop, this factor is the same for all terms. If the common factor is g , then

$$(a, r, n) = g\left(\frac{a}{g}, r, \frac{n}{g}\right).$$

In the remaining analysis it is assumed that t_i and n have no common factor. It follows that a and n have no common factor, and that r and n have no common factor.

The next step is to reduce the modulus if it is not a power of a prime. If the modulus is not a power of a prime, then it is the product of two relatively prime factors, say mn . From number theory it is known that there exist integers x and y such that $mx + ny = 1$. It follows that there are integers x modulo n and y modulo m such that

$$mx + ny = 1 \text{ mod } mn.$$

If we operate on the progression

$$(a, r, mn)$$

as follows, the progression is simplified

$$\begin{aligned}(a, r, mn) &= (mxa + nya, r, mn) \\ &= (mxa, r, mn) + (nya, r, mn) \\ &= (m(xa, r, n) + n(ya, r, m))\end{aligned}$$

These progressions have smaller moduli. This kind of reduction can be repeated until the modulus is a power of same prime, p .

The analysis of a progression whose modulus is p^n is carried out in p -ary notation, that is, in a number system in which there are p digits and in which the sequence $d_n d_{n-1} \dots d_0$ means $\sum_{i=0}^n d_i p^i$. (Example: 2-ary means binary.) In this notation, the expression "modulo p^n " means simply "saving only the n least significant digits." We shall refer to such a progression as an n -digit p -ary progression.

There are two kinds of operations which can be used to simplify the study of n -digit p -ary progressions. They are:

1. Contraction. Replacing the progression (a, r, p^n) by (a, r, p^{n-k}) . This is equivalent to dropping the first k p -ary digits from consideration.
2. Extension. The inverse of contraction. Replacing the progression (a, r, p^n) by (a', r', p^{n+k}) where $a' = a \bmod p^{n+k}$ and $r' = r \bmod p^{n+k}$. This is equivalent to adding k initial p -ary digits.

An extension or contraction is simple if $k=1$. Note that a simple contraction is a unique operation, but simple extension is not, since the p -ary digits annexed at the left of a and r are arbitrary.

Example: The 2-digit 3-ary progression $(7, 5, 3^2)$ has the simple contraction $(1, 2, 3)$, since, in 3-ary,

7 is written 21 and

5 is written 12.

The 1-digit 3-ary progression $(1, 2, 3)$ has, on the other hand, nine simple extensions; a can be changed to 1, 4 or 7, (in 3-ary, 01, 11, or 21) and r can be changed to 2, 5 or 8 (in 3-ary, 02, 12 or 22), with n changed from 3 to 3^2 .

It is clear that a simple extension of a progression multiplies the period by a factor not less than one and not more than p . The rather surprising fact is that every simple extension is either minimal or maximal; that is, it either leaves the period unchanged or multiplies it by p . There are some additional rules applicable to maximal and minimal extensions, and these rules each have one exception.

1. Every extension of a maximal extension is maximal.

Exception: $(a, 3, 2^2)$ although a maximal extension of $(a, 1, 2)$ has only minimal extensions.

2. There is exactly one simple extension of the multiplier in a minimal extension which produces another minimal extension; all other simple extensions of the multiplier produce maximal extensions of the progression.

Exception: The simple extension, 3 modulo 2^3 of the multiplier 3 modulo 2^2 produces minimal extensions of progression but it, in turn, has only maximal extensions.

The following diagram of 3-ary progressions and their periods will illustrate the rules above. Each solid arrow represents a maximal extension and each dotted arrow a minimal extension. The number under each progression is its period. The number above is the multiplier in p-ary.

To determine the period of an n-digit p-ary progression, one first determines the period of the last digit. This is always a divisor of p-1,

and if the last digit of the multiplier is properly chosen, the period is $p-1$. There seems to be no simple rule better than trial and error for choosing the multipliers which produce a period of $p-1$.

Having determined the period of the last digit, one can find the period of the progression by considering successive simple extensions, and by using the rules stated above. Again, there seems to be no rule simpler than trial and error for determining which of the simple extensions of a minimal extension is a minimal extension.

If the object is to produce a maximum period, then there are at most two choices (the last two digits) or, in the case of $p=2$, the last three. Furthermore, if the period is to be maximal, then all acceptable choices of starter and multiplier produce progressions which include all n digit p -ary numbers except those ending in zero, with the starter and multiplier determining only the order. The exception to the preceding is again the case $p=2$, in which case the numbers generated are all n -digit binary numbers having two of the last three binary bits specified (i.e., ending in octal (1 or 3), (5 or 7), (1 or 5) or (3 or 7)). It is clear that the distributions of these numbers leave little to be desired, being completely regular and uniform except in the last digit for $p \neq 2$, and the last three bits for $p=2$.

What is not determined is the order in which numbers come up. Apparently no general statement can be made, for example, on the distribution of the first million numbers in the 16 billion numbers in $(1, 3, 2^{36})$. At this

point, we are reduced to the drudgery of experiment. Several tests for random numbers have been applied to these series, and to date no significant deviation from randomness has been detected.

Some special consequences for octally or decimally represented numbers may be of interest. The maximum period for an n -digit binary generator is 2^{n-2} , produced by any odd starter and any multiplier whose last octal digit is 3 or 5. The numbers generated are all those whose final octal digit is (1 or 5), (3 or 7), (1 or 3) or (5 or 7), depending on the starter and multiplier. For an n -digit decimal generator, the problem is a little more complicated, since the first step in simplification requires a separation into two progressions with moduli of 2^n and 5^n respectively. After analyzing these, the answers can be recombined to produce results stated in decimal. The maximum period is $5 \times 10^{n-2}$ *, produced by any starter whose last decimal digit is 1, 3, 7, or 9 and any multiplier whose last three digits end with

003,	013,	027,	037,	053,	067,	077,	083,
117,	123,	133,	147,	163,	173,	187,	197

or any of these numbers increased by a multiple of 200. Such a progression will generate all integers which are congruent to the first four terms modulo 80.

* For $n \geq 4$. For lower powers of 10, the periods are 10, 4; 100, 20; 1000, 100.

Issued by
Technical Information Division
Sandia Corporation
Albuquerque, New Mexico