Technical Report CS74014-R

SOME EXPERIMENTAL OBSERVATIONS ON THE BEHAVIOR OF COMPOSITE RANDOM NUMBER GENERATORS†

Richard E. Nance
and
Claude Overstreet, Jr.*

Department of Computer Science
College of Arts and Sciences
Virginia Polytechnic Institute and State University
Blacksburg, Virginia 24061

Revised
August 1976

†Research supported in part by a grant to Claude Overstreet, Jr. by the Department of Computer Science, V.P.I. and S.U. Both authors also wish to acknowledge that early work using an SRU 1108 was conducted at the Department of Computer Science and Operations Research, Southern Methodist University.

*Department of Computer Science, Bowling Green State University, Bowling Green, Ohio.

ABSTRACT

A series of experiments with composite random number generators utilizing shuffling tables is described. The factors investigated are: (1) the magnitude of the modulus (equivalently, the word-size of the machine), (2) the effect of the modulus value for the indexing generator, and (3) the table size used for shuffling. Experimental results indicate that:

(1) on large word-length machines (permitting large modulus values), shuffling accomplishes little in comparison with selected simple generators,

(2) on small word-length machines, shuffling can produce sequences having an increased period and demonstrating acceptable statistical behavior, and

(3) a table size of 2 produces results comparable to those obtained with larger tables.

Keywords: random number generation, composite methods, increased periodicity, sample size, table size

Computing Reviews categories: 5.39

INTRODUCTION

The Role of Random Numbers

Random number generation represents a research intersection of four disciplines: computer science, discrete mathematics, operations research and statistics. Interestingly, each discipline contains researchers who contribute to the development of random number generation as well as those who utilize random numbers in other research areas. Digital computer simulation is perhaps the most apparent area of overlap between computer science and operations research, and random number generation occupies a major role in computer simulation. However, the use of random numbers by experimentalists in both disciplines extends quite prominently into other areas:

(1) Monte Carlo techniques in numerical analysis,

(2) generation of test samples for experimentation in mathematical programming, and

(3) hash coding methods in information structures and data base design.

Regarding the development of random number generation, researchers in computer science and operations research (and in discrete mathematics and statistics) have devoted considerable attention to the various subtopics. The books of Knuth [4], Jansson [3], Newman and Odell [12] and the chapter by Page [14] are examples of this attention. Yet, despite these works and those of a large number of the 491 references in Nance and Overstreet [10], much dissatisfaction is expressed regarding the known techniques for random number generation [5, 8].

Background of Composite Methods

The motivation for composite, or combinational, random number generators stems from a perceived need to improve the statistical properties of sequences produced by the linear congruential method of Lehmer [6]. This necessary improve-

ment is especially crucial according to those who have investigated the additive techniques [2,9]. With respect to the methods employing multiplication, i.e.

$$X_{n+1} \equiv aX_n + c \pmod{m} \quad ,$$

the intended improvement would result from the reduction of dependency. Intuitively, by "shuffling" the sequence, the successor and predecessor residues would no longer be adjacent, and the sequence would "seem to be" comprised of independent members. However, as Knuth [4, pp. 4-5] illustrates, intuition can be quite deceptive with random number generation.

MacLaren and Marsaglia [7] introduced the table procedure for shuffling members of a sequence in seeking to derive an acceptable generator from two congruential generators that individually proved unacceptable. The two generators

$$U_{k+1} \equiv (2^{17} + 3)\, U_k \pmod{2^{35}} \qquad (1)$$

$$V_{k+1} \equiv (2^{7} + 1)\, V_k + 1 \pmod{2^{35}} \qquad (2)$$

were used in conjunction with a table of 128 locations. Generator (1) was used to produce values, and generator (2) produced location values to establish the sequence ordering of values from (1). The composite generator realized a definite improvement in statistical behavior.

Although later research by Van Gelder [15] suggested that the poor behavior of the individual congruential generators (we use the term "simple generators") stemmed from poor choices of multipliers rather than inherent failings, the composite generator became quite popular. A recent bibliography [10] contains nine references treating composite generators as the primary subject. Although composite generators can have extremely long periods, we suspect that the popularity of the composite method is due to an intuitive belief that the generated sequence must reflect a greater degree of stochastic independence, which is not necessarily the case.

Learmouth and Lewis [5] showed that a generator exhibiting poor behavior - the RANDU generator from the IBM Scientific Subroutine Package - can be improved to an acceptable level, generally speaking, by the table shuffling method. During the process of their experimentation, they noted the lack of power of the runs test in discriminating poor generators.

An interesting composite method, proposed by Westlake [16], uses a bitwise addition modulo 2 of the results from two simple generators, with the value from one generator cyclically permuted by a "random" number of positions. In this paper we concentrate on composite generators based on the shuffling table procedure, and we do not comment on generators of the Westlake type.

The distinct disadvantage of the composite generator is the cost in terms of execution time.[1] In a previous paper [11] we present experimental evidence that the composite generator requires slightly less than twice the execution time of a simple generator on a SRU 1108 using the FORTRAN language. The crucial question is: "Does the improvement in statistical behavior warrant the additional cost in execution time?" We seek to answer this question in a subsequent section.

Test Environment

Several factors can affect experiments with random number generators, some of which are:

(1) machine architecture - the word size, the use of one's- or two's-complement arithmetic, the instruction set;

---

[1] Composite generators also require more storage than simple generators; however, the additional storage is not so costly with contemporary machines as with early computers.

(2) language - conditions on overflow causing interrupts, implementation of integer multiplication, presence of logical operators, bit manipulation capabilities; and

(3) operating system - particularly in its space and time accounting when space or time comparisons are attempted.

We mention these factors simply because the experimenter can exert little or no control over them; he must deal with each one in a way that maintains the validity of his experiment.

The experiments described in this paper present results using two different computer systems:

(1) An SRU 1108 (36-bit word and one's-complement arithmetic) under an EXEC 2[2] operating system was used for the comparison of simple and composite generators on large word-length machines. The generators were programmed in FORTRAN V, level 3.4.

(2) An IBM S/370 Dual 158 (32-bit word and two's-complement arithmetic) under OS/ASP-MVT was used for the investigation of composite generators on small word-length machines. The generators were programmed in FORTRAN IV, level H, in which the small word-length was simulated.

Statistical tests were accomplished using the test package for random number generators, described in a previous report [13]. This package contains programmed versions of all the tests described by Knuth [4]. Results in some cases were summarized further to facilitate interpretation and comparison.

DESCRIPTION OF EXPERIMENTS

General Test Procedures

An important consideration in the use of random number generators is the sample size, i.e. the number of sequence values to be required for the experiment. Our experience indicates that the sample size should not exceed one-fourth of

[2] A UNIVAC supported operating system.

the period. To present persuasive arguments for this "rule of thumb" would occupy too much space, but Figure 1 is indicative of the behavior that is realized for the coupon collector's test (see [4, p. 58-59]). The results shown in Figure 1 are the χ^2 values resulting from the comparison of expected and observed sample sequences for the coupon collector's test. In this case the periodicity of the generators used

$$X_{n+1} \equiv (181 + 16J)\ X_n + 45 \pmod{2^{11}} \qquad J=0,\ 1,\ \ldots,\ 9$$

is 2048, and the ten generated sequences for each sample size represent approximate proportions of .25, .50, .75 and 1.0 of the sequence (the population so to speak). An obvious shift in the generation of sequences failing the coupon collector's test is shown in the four histograms of Figure 1. As the sample includes a higher proportion of the period, the number of failing sequences increases.

In the remainder of this paper we investigate the following characteristics of composite generators:

(1) periodicity,
(2) statistical behavior compared with simple generators for large word-length machines (large modulus),
(3) effect of the modulus value used for the indexing generator (the generator producing table locations), and
(4) effect of table size.

The last two characteristics are limited to small word-length generators because of the results of (2) above.

The Effect of the Modulus Value on Periodicity

Consider the composite generator

$$Y_{n+1} \equiv a_1\ Y_n + c_1 \pmod{m_1}$$

$$Z_{n+1} \equiv a_2\ Z_n + c_2 \pmod{m_2}$$

where we assume that {Z} is the sequence of table locations from which values of {Y} are selected. According to a well known theorem (for example, see [3, 48-49]),

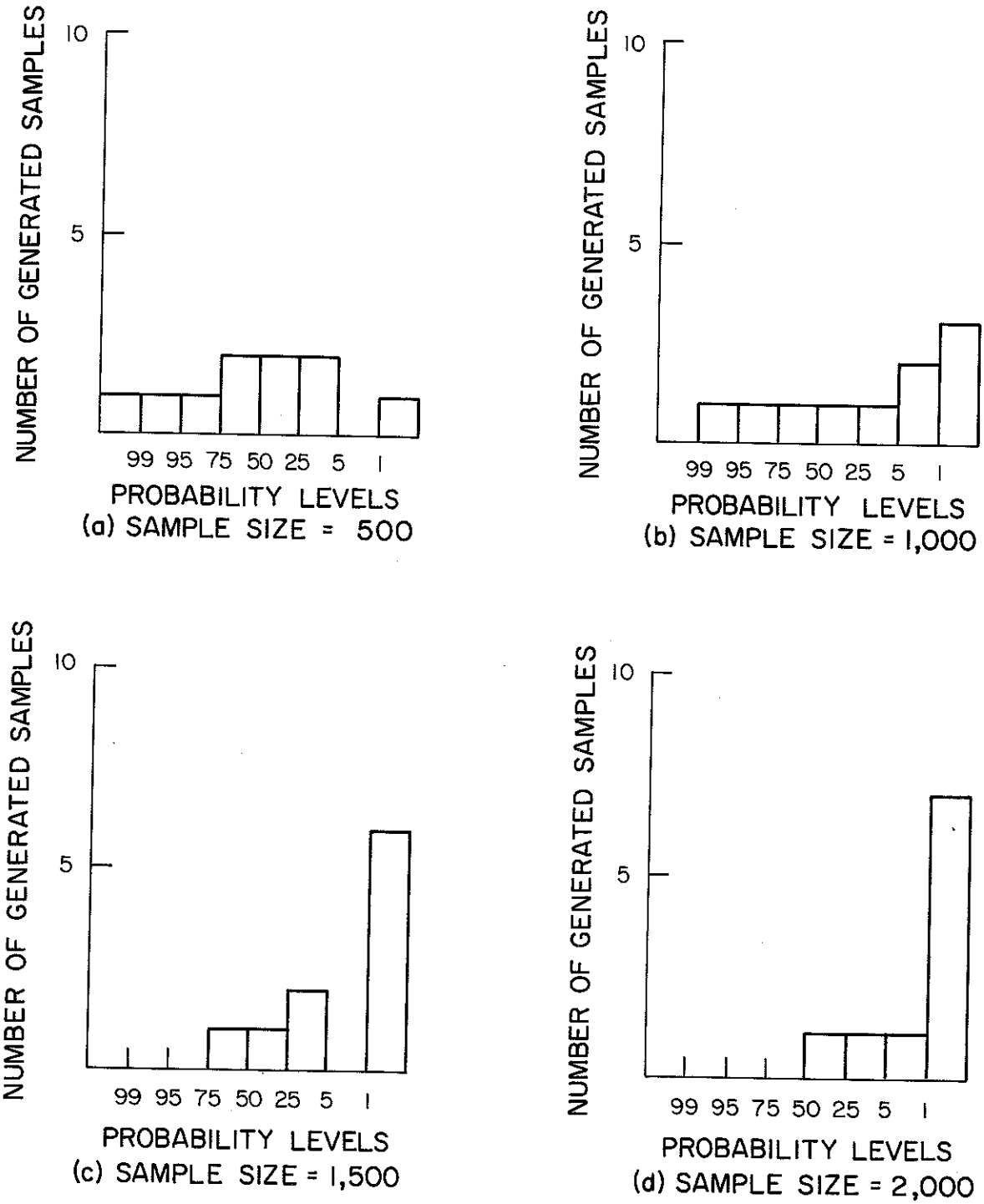

Figure 1. Effect of Increasing Sample Size
on Coupon Collector Test Results

if (m_1, m_2) are relatively prime and $H(m_1)$, $H(m_2)$ are the respective periods of the above generators, then

$$H(m_1 \cdot m_2) = \text{lcm}\,[H(m_1), H(m_2)]$$

where $H(m_1 \cdot m_2)$ is the period of the composite generator.

To obtain the maximum period for a composite generator, we must select m_1 and m_2 relatively prime so that $H(m_1)$ and $H(m_2)$ are maximum and lcm $[H(m_1), H(m_2)] = H(m_1) \cdot H(m_2)$. This of course prohibits <u>both</u> m_1 and m_2 from being powers of 2, for if $m_1 = 2^d$ and $m_2 = 2^e$, then the period of the composite generator is the maximum of the two, i.e.

$$H(m_1 \cdot m_2) = \max\,[H(m_1), H(m_2)].$$

Obviously, since $H(2^d) = 2^d$ for $d \geq 3$, the selection of any prime ≥ 7 effects an increased period. Keep in mind that nothing discussed to this point relates to the table size, merely the generator used to produce table locations.

Statistical Behavior of Simple and Composite Generators on Large Word-Length Machines

With a large word-length and a large modulus, little need would normally exist for constructing the composite generator so as to increase periodicity. Recognizing the added cost of using a composite generator supposedly to improve the independence of sequence values, we must question if the anticipated improvement in statistical behavior warrants the additional cost.

In an effort to answer this question, we use a simple generator with three different multiplier values, each of which exhibits comparatively good behavior when subjected to the spectral test as reported by Knuth [4, p. 88].

$$a = \begin{cases} 5^{15} & \text{(the first simple generator)} \\ 2718281821 & \text{(the second simple generator)} \\ 3141592221 & \text{(the third simple generator)} \end{cases}$$

$$X_{n+1} \equiv aX_n \pmod{2^{35}}$$

and a composite generator

$$Y_{n+1} \equiv 5^{15} Y_n \pmod{2^{35}}$$

$$Z_{n+1} \equiv 3141592221 Z_n \pmod{2^{35}}$$

where the six most significant bits of members of {Z} are used to generate locations in a table of size 128. A sample of 5,000 values is produced for each of the three simple generators and the composite generator. The samples are submitted to nine tests included in the random number generation test package. The specific tests used are the chi-square with two different subsequence lengths (200 and 1000), correlation of values with lags from one to twenty, coupon collector, gap over the intervals [0.4,0.5], (0.0,0.5) and (0.5,1.0), permutation, poker, runs up, serial, and the Kolmogorov-Smirnov (K-S) goodness-of-fit. Results are presented in Table 1.

In comparing two distinct methods, we prefer the rather cluttered presentation of Table 1 to a simpler summarized description of test results. Our preference stems from the belief that acceptable behavior of a random number generator, like beauty, "lies in the eyes of the beholder". Table 1 permits the reader to develop his conclusions regarding the two methods. In subsequent experimentation we use a more condensed summary of the effects on the composite generator produced by choice of modulus and table size.

Viewing the test results in Table 1, one has difficulty in substantiating the argument that the composite generator should evince statistical behavior superior to that of the simple generators. Following the suggested procedure of Knuth [4, pp 39-40] we consider any observed values falling outside the probability interval [.05, .95] to be "suspect", and from these values, any falling beyond the interval [.01, .99] are termed "bad".

TESTS	Generators Tested															
	IA = 5**15 IX = 56329 ONE-LINE 1				IA = 3141592221 IX = 56329 ONE-LINE 2				IA = 2718281821 IX = 56329 ONE-LINE 3				IA=5**15, IAJ=3143592221 IX = 56329, IXJ = 56329 COMPOSITE			
1. Chi-Square: (10 intervals of 10 values)	.513	.519	14.4	9.6	.482	.502	9.4	6.4	.514	.547	4.6	9.0	.482	.486	18.4	10.0
	.506	.489	6.4	9.0	.494	.474	8.6	18.6	.435	.493	17.4	4.4	.483	.489	7.4	1.4
	.513	.491	7.8	2.4	.515	.516	7.2	15.4	.491	.491	16.6	7.6	.556	.449	6.8	9.8
Uniformity	.509	.443	2.8	10.6	.436	.535	10.2	10.8	.504	.440	8.6	6.4	.540	.463	17.2	5.2
	.554	.469	11.6	7.2	.514	.483	8.8	8.8	.553	.514	9.2	9.6	.523	.476	6.4	3.4
Chi-Square: (10 intervals of 100 values)	.500	.495	6.8	10.1	.495	.490	20.5	12.0	.498	.497	9.4	5.9	.494	.492	6.4	10.9
	.490	.506	9.6	8.4	.490	.514	8.5	12.3	.496	.502	2.5	13.0	.492	.510	11.1	11.0
	.499	.512	9.5	4.1	.484	.487	19.2	9.9	.498	.519	5.9	9.2	.499	.506	7.6	5.2
Uniformity	.498	.505	8.3	8.0	.491	.511	5.1	6.4	.499	.497	7.0	7.3	.502	.506	7.6	7.1
	.496	.491	1.4	13.7	.499	.494	10.2	11.3	.576	.483	16.9	11.8	.496	.497	1.6	9.5
Probability Level			99%		95%		75%		50%		25%		5%		1%	
χ^2 Value (9 d.f.)			2.088		3.325		5.899		8.343		11.390		16.920		21.670	
2. Correlation: Values are listed in the following order, Lags 1-5, 6-10, 11-15, 16-20	.026	-.006	-.007	-.010	-.012	.001	-.008	-.002	.028	.018	.006	-.011	-.018	-.014	-.001	.025
	-.003	.025	.012	.022	.008	-.014	-.025	.017	.023	.008	.020	.004	.021	.006	-.008	-.008
	.038	-.002	.003	.021	-.008	.014	.029	-.014	.015	.011	.013	-.023	.002	-.016	.001	.012
	-.008	.001	-.028	.029	.006	-.014	.013	-.018	.007	.016	.003	.004	.009	.023	-.037	.006
	.003	-.004	.015	-.019	-.014	-.007	.008	.001	.017	-.007	.010	-.036	.012	.002	-.016	.012
3. Coupon Collector: Segments Requested/Found	200/200				200/200				200/200				200/200			
Maximum Segment Length Found	31				38				35				28			
χ^2 Value	8.8562				16.0881				17.3772				21.7633			
Probability Level			99%		95%		75%		50%		25%		5%		1%	
χ^2 Value (14 d.f.)			4.119		6.305		10.220		13.500		17.235		23.401		28.291	
4. Gap Test: Specified Gap Interval	0.4-0.5	0.0-0.5	0.5-1.0		0.4-0.5	0.0-0.5	0.5-1.0		0.4-0.5	0.0-0.5	0.5-1.0		0.4-0.5	0.0-0.5	0.5-1.0	
Gaps Requested/Found	200/200	200/200	200/200		200/200	200/200	200/200		200/200	200/200	200/200		200/200	200/200	200/200	
Maximum Gap Length	39	10	7		51	9	9		50	8	8		45	9	4	
χ^2 Value	19.593	7.190	2.480		9.238	2.380	0.540		9.610	6.660	4.420		10.551	1.840	2.620	
(d.f.)	(13)	(4)	(4)		(13)	(4)	(4)		(13)	(4)	(4)		(13)	(4)	(4)	
Probability Level			99%		95%		75%		50%		25%		5%		1%	
χ^2 Value (13 d.f)			3.575		5.628		9.353		12.500		16.103		22.078		26.835	
χ^2 Value (4 d.f)			.297		.711		1.923		3.357		5.385		9.488		13.280	
5. Permutation: (1666 Groups of 3 elements)																
χ^2 Value (5 d.f)	6.9064				4.9112				4.3782				6.6903			
Probability Level			99%		95%		75%		50%		25%		5%		1%	
χ^2 Value (5 d.f)			.554		1.145		2.675		4.351		6.626		11.070		15.090	

Table 1. Statistical Test Results for the Three One-Line Generator and the Composite Generator

6. Poker: (For a group of 5 numbers, predicting the number of unique values from 1 to 10)		χ^2 = .3748		χ^2 = 3.9145		χ^2 = 1.7195		χ^2 = .9398
Number Different	Predicted							
2	13	15		10		9		12
3	179	180		180		183		187
4	503	509		482		509		509
5	302	295		327		298		291
Probability Level		99%	95%	75%	50%	25%	5%	1%
χ^2 Value (3 d.f.)		.115	.352	1.213	2.366	4.108	7.815	11.340
7. Runs Up:								
χ^2 Value		12.3740		3.3508		6.5132		6.1828
Maximum Run Length		6		6		7		7
Probability Level		99%	95%	75%	50%	25%	5%	1%
χ^2 Value (6 d.f.)		.872	1.635	3.455	5.348	7.841	12.590	16.810
8. Serial: (Testing pairs of numbers (Q,R) where Q and R lie between 0 and 2)		χ^2 = 7.6122		χ^2 = 7.8139		χ^2 = 6.9496		χ^2 = 6.1828
Probability Level		99%	95%	75%	50%	25%	5%	1%
χ^2 Value (8 d.f.)		1.646	2.733	5.071	7.344	10.220	15.510	20.090
9. Kolmogorov-Smirnov: (Testing 4 sets of 500 values)		K+(1)=.633 K-(1)=.587		K+(1)=1.546 K-(1)=.642		K+(1)=.698 K-(1)=.446		K+(1)=.131 K-(1)=.975
		K+(2)=.659 K-(2)=.684		K+(2)=1.121 K-(2)=.208		K+(2)=1.077 K-(2)=.946		K+(2)=1.215 K-(2)=.175
		K+(3)=.583 K-(3)=.645		K+(3)=1.006 K-(3)=.252		K+(3)=.646 K-(3)=.374		K+(3)=.565 K-(3)=.723
		K+(4)=.661 K-(4)=.467		K+(4)=.557 K-(4)=.655		K+(4)=.567 K-(4)=.930		K+(4)=1.153 K-(4)=.185
Probability Level		99%	95%	75%	25%	5%	1%	
K-S Values		.0642	.1538	.3726	.8254	1.2163	1.5085	
Application of K-S test to previous K-S values.		(K+)+=.835 (K+)-=.986		(K+)+=.017 (K+)-=1.24		(K+)+=.254 (K+)-=.948		(K+)+=.433 (K+)-=.860
		(K-)+=.785 (K-)-=.707		(K-)+=.848 (K-)-=.166		(K-)+=.343 (K-)-=.645		(K-)+=.867 (K-)-=.297
Probability Level		99%	95%	75%	25%	5%	1%	
K-S Values		.0194	.0879	.3202	.7642	1.1280	1.3780	

Table 1 - Continued

Little difference in the test results appear for the coupon collector, permutation, poker, runs up, and serial tests. For the gap test on the intervals [0.4,0.5] and (0.0,0.5) little difference in behavior is apparent. However, for the second simple generator the gap test on the interval (0.5,1.0) produces a χ^2 value of .540 which is in the "suspect" range but not "bad". The χ^2 test of uniformity shows both the second and third simple (or one-line) generators to produce a single "suspect" sequence among the ten sequences of 10 values each. (Note that by applying this test to sequences of only 10 values, we make this test rather severe.) The first simple generator produces two "suspect" sequences, and the composite generator produces two "suspect" and one "bad". With ten sequences of 100 values, the first simple and the composite generator produce one "bad" sequence each; while the second produces two "suspect" sequences, and the third, one "suspect" sequence.

The Kolmogorov-Smirnov (K-S) test of uniformity shows the second one-line generator to produce one "bad" sequence and the composite, one "suspect" sequence. Application of the K-S test to the computed K-S statistics indicates the second simple generator to produce one "bad" and one "suspect" sequence.

We include the correlation coefficients (for lags 1 to 20) without an explicit test. We could apply the distribution test of the coefficient, developed by Anderson [1] and described in Jansson [3, p. 140-143]. Personally, finding none of the 20 lagged correlation coefficients outside the interval (-0.05,0.05) seems encouraging, but we might be slightly concerned that the third one-line generator produces only four negative values.

Our conclusion is that the composite generator, when compared with "good" simple generators, demonstrates no better performance.[3] However, this conclusion must be limited to cases where relatively large word-lengths (large modulus

[3] Learmonth and Lewis [5] have shown that a poor generator (RANDU) can be improved by shuffling. We begin with simple generators exhibiting good behavior.

values) enable relatively large periods. With small word-length machines (the 12- and 16-bit minicomputers), the composite generator offers two potentials:

(1) the combination of unacceptable simple generators to yield a more acceptable sequence, and

(2) the realization of sufficient periodicity to enable reasonable sample sizes.

To evaluate these potential capabilities, we conduct the remaining experiments assuming a small word-length limitation on the modulus value.

Experiments with Generators Having a Limited Modulus Value

In the remaining discussion we use a mixed generator to produce the uniform values on the interval (0,1), referring to it as the primary generator. A second generator, the indexing generator, identifies table locations from which previous values produced by the primary generator are taken and into which new values of the primary generator are placed. The modulus of the primary generator is 2^{11} for all cases.

Effect of Composite Method on Statistical Behavior

Let us consider the primary generators

$$Y_{n+1} \equiv (181 + 16J)\, Y_n + 45 \pmod{2^{11}} \qquad J = 0,1, \ldots, 9$$

From our earlier discussion we note that selection of a modulus of 2^b, $3 \leq b \leq 11$, for the indexing generator, does not extend the period of the composite generator. However, can we discern improved statistical behavior for the composite generator over the primary generators used above? We use as the indexing generator

$$Z_{n+1} \equiv 125\, Z_n + 31 \pmod{2^{11}}$$

which, in earlier testing of a sample of 500 values, had yielded no "bad" values and only three "suspect" values (all on the K-S test). The results for the

composite and the primary generator used alone are displayed in Figure 2. Also shown in Figure 2 are the test results for a composite generator, using as the indexing generator

$$\hat{Z}_{n+1} \equiv 13\ \hat{Z}_n \pmod{151}$$

A table size of 32 is employed in both composite generators.

Figure 2 provides a summary of test results based on a probability level (essentially, a probability statement). The reader should interpret the probability level x as representing a value that would be exceeded by a truly random sequence with probability .x. For example, the level 25 represents a χ^2 value that would be exceeded by a truly random sequence with probability .25. In all cases a sample of 500 values is generated by each of the ten generators. For both the chi-square and K-S tests, each sample of 500 is divided into five subsamples of 100 each, so as to improve the discrimination. Consequently, 50 subsequences are evaluated for the chi-square and K-S tests, and ten for each of the remaining tests.

The coupon collector, gap, permutation, poker, runs and serial tests require a frequency count among categories followed by a χ^2 statistic based on the relative difference in theoretical and actual frequencies. Consequently, the probability levels admit the same interpretation with these tests, and the summary in Figure 2 is obtained.

Continuing the evaluation procedure suggested by Knuth, we note that both composite generators seem to cause slight improvements. While a total of 11 "bad" sequences are produced by the primary generator alone, six are produced by the composite generator (mod 151) and only three by the composite generator (mod 2^{11}). We might be tempted to believe that, although not increasing the period, the composite generator (mod 2^{11}) does exhibit improved statistical behavior.

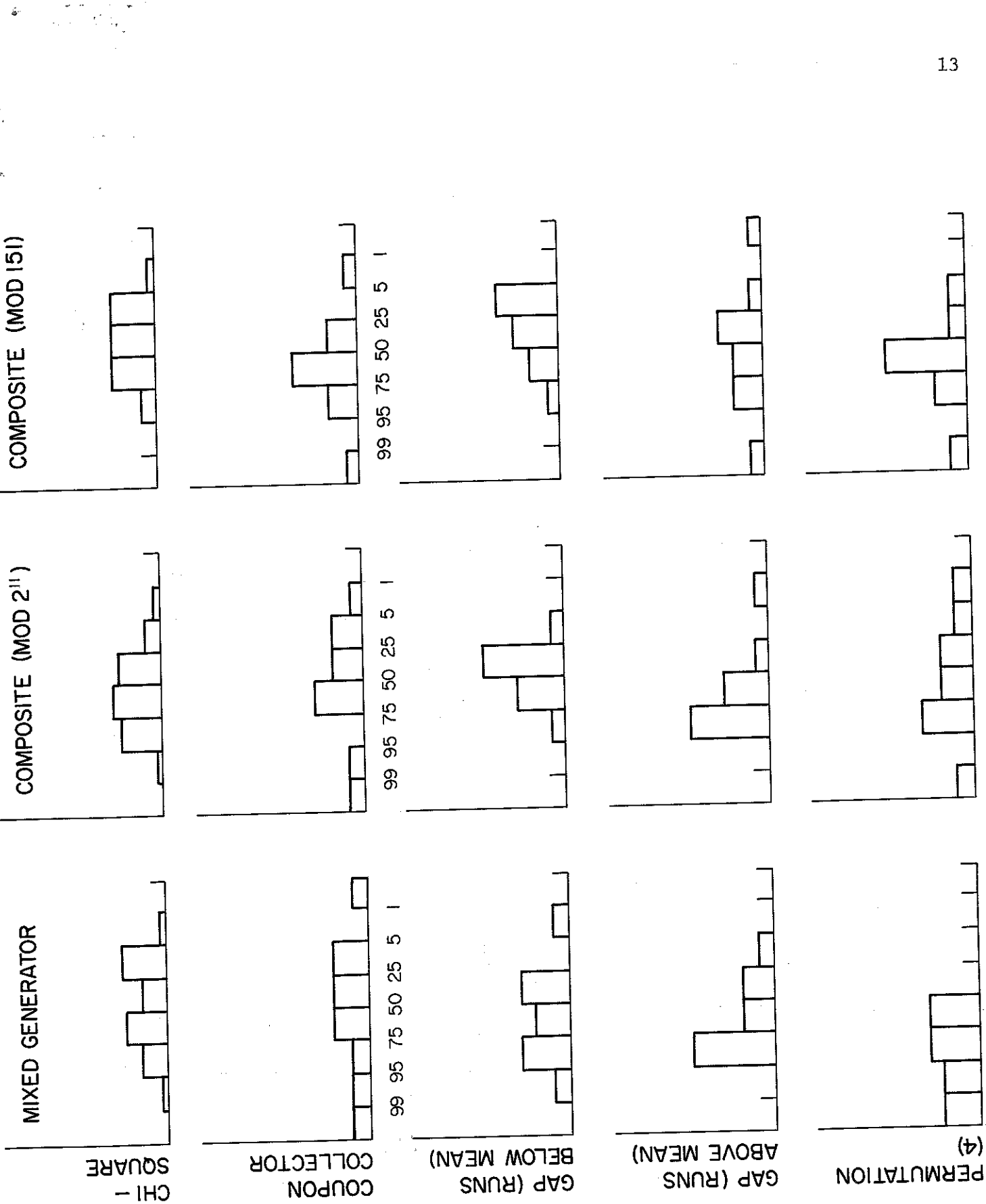

Figure 2. Comparison of Test Results for Mixed, Composite Generator (mod 2^{11}) and Composite Generator (mod 151)

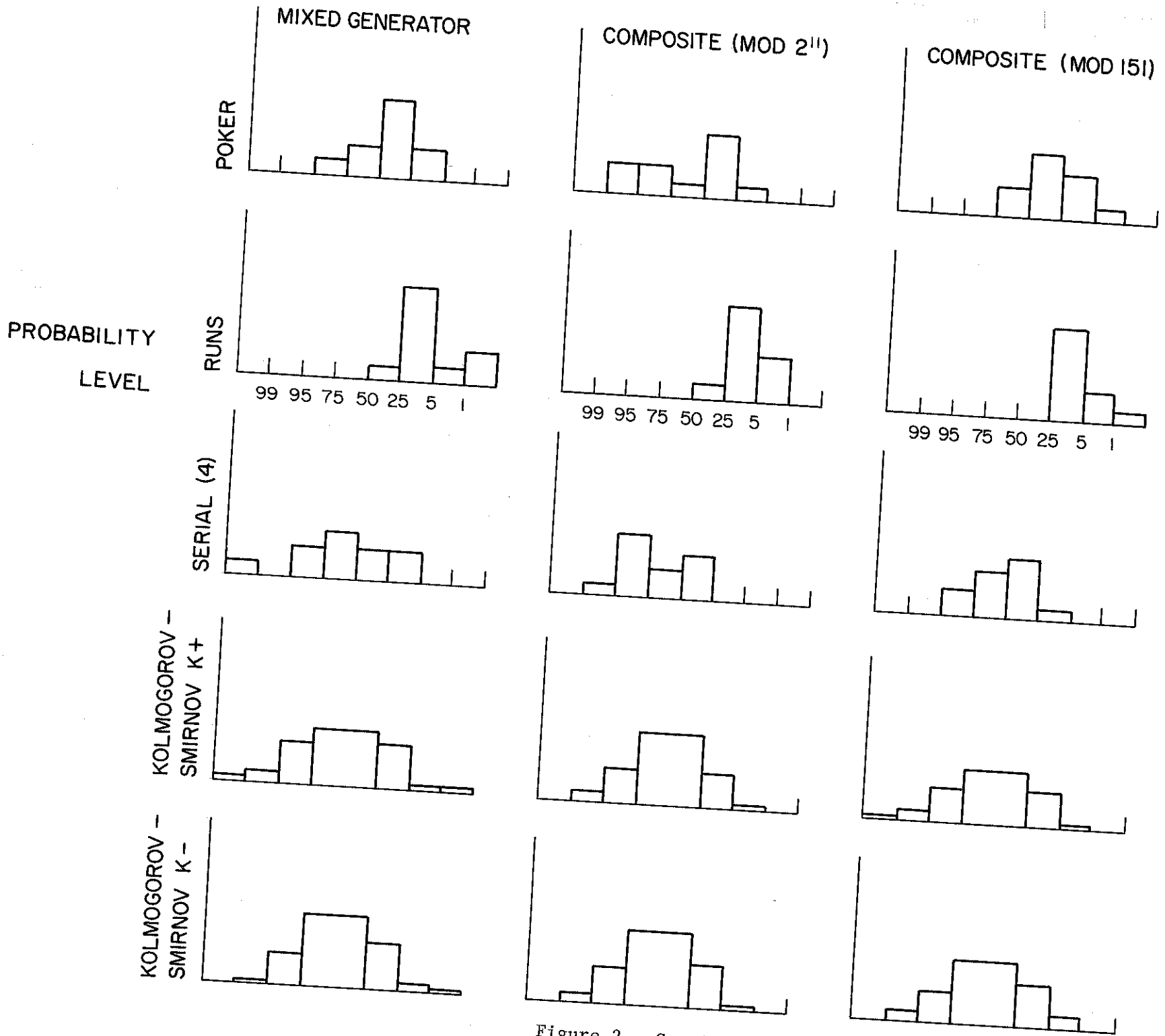

Figure 2. Continued.

However, we see that this generator produces 24 "suspect" sequences while the composite generator (mod 151) produces 18. The primary generator alone produces 19 "suspect" sequences. The proper conclusion seems to be that no definitive improvement can be claimed for shuffling.

Effect of Increased Periodicity on Statistical Behavior

Having observed that apparently shuffling realizes little benefit if the sample size (for experimental use) is well within the period of the primary generator, we now ask: "Can shuffling be used as an effective procedure for obtaining an acceptable generator of greater period than the primary generator?" As we note above, we must select the modulus for each generator such that the prime factors of each modulus are not mutually divisible. We select the following:

$$Y_{n+1} \equiv (181 + 16J)\, Y_n + 45 \pmod{2^{11}} \qquad J = 0,1, \ldots, 9$$

$$Z_{n+1} \equiv 13\, Z_n \pmod{151}$$

where the first is the primary generator. Use of a multiplicative generator for the index is prompted by the desire to eliminate one addition operation since the period of the composite should be quite sufficient.

The period of this composite generator is readily determined to be 153,600 by noting that

$$H(2^{11}) = 2048 \qquad H(151) = 150$$

$$H(2048,150) = \text{lcm}\,(2^{11},2,3,5^2)$$

$$= 75(2^{11})$$

A comparison of the statistical behavior of the primary generator alone with the composite generator using a table size of 32 is provided by Table 2. The improvement realized by shuffling is pronounced. Ignoring the chi-square and

	Probability Level						
	99	95	75	50	25	5	1
Chi-Square	7 8	13 15	20 16	1			
Coupon Collectors			3	1	1 3	1 3	1	7
Gap (Runs Below Mean)	3	2 1	3 3	1	1 3	1	1	1
Gap (Runs Above Mean)	4	2 1	1 3	2 2	2	2		1
Permutation (4)	7	1	3	2 3		2	2	
Poker	1		4 1	1	4	4 4	1	
Runs	1	1	1 3	1 1	4 3	1	1	1 2
Serial (4)	10 1	2	2	3	1	1		
Kolmogorov - K+		2	9	21		9	1	
Smirnov K-	1 1	2 2	11 8 9	15 19 17		12 10 10	1 1	

Table 2. Effect of Composite Method on Statistical Behavior (Upper value is for primary generator alone, lower value, for composite generator.)

K-S tests, only three "bad" sequences are produced by the composite generator; while 36 "bad" sequences result from the primary generator used alone. The composite generator produces seven "suspect" sequences, and the primary generator produces nine.

Our motivation for excluding the chi-square and Kolmogorov-Smirnov tests from the above discussion is prompted by the fact that both are tests of uniformity. Shuffling is likely to have but little effect on uniformity, and this is seen in Table 2 where only slight changes are observed for the composite generator. However, shuffling brings about decided improvement in the results for the coupon collector, gap (both runs below and above the mean), permutation, and serial tests. Slight improvement or no worse behavior is indicated by the poker and runs tests.

Effect of Table Size on Statistical Behavior

The pronounced benefit of shuffling with an increased period prompts another consideration: the ability to reduce the overhead associated with the shuffling table without impairing the statistical behavior. The amount of overhead is directly proportional to the size of the shuffling table; consequently, we experiment with table sizes of 2, 4, 8, 16, and 32. Only the results for 2, 8 and 32 are presented since the other values provide no additional information.

Figure 3 contains a series of four histograms for each of seven tests. From left to right, the histograms present the results for (a) the primary generator alone and the composite generator with table size of (b) 2, (c) 8, and (d) 32. We do not present results for the chi-square and K-S tests simply because, as we have noted, both are tests of uniformity.

The most surprising result of the experiment is that a table size of 2, in general, precipitates results equal to those for a table size of 32. This

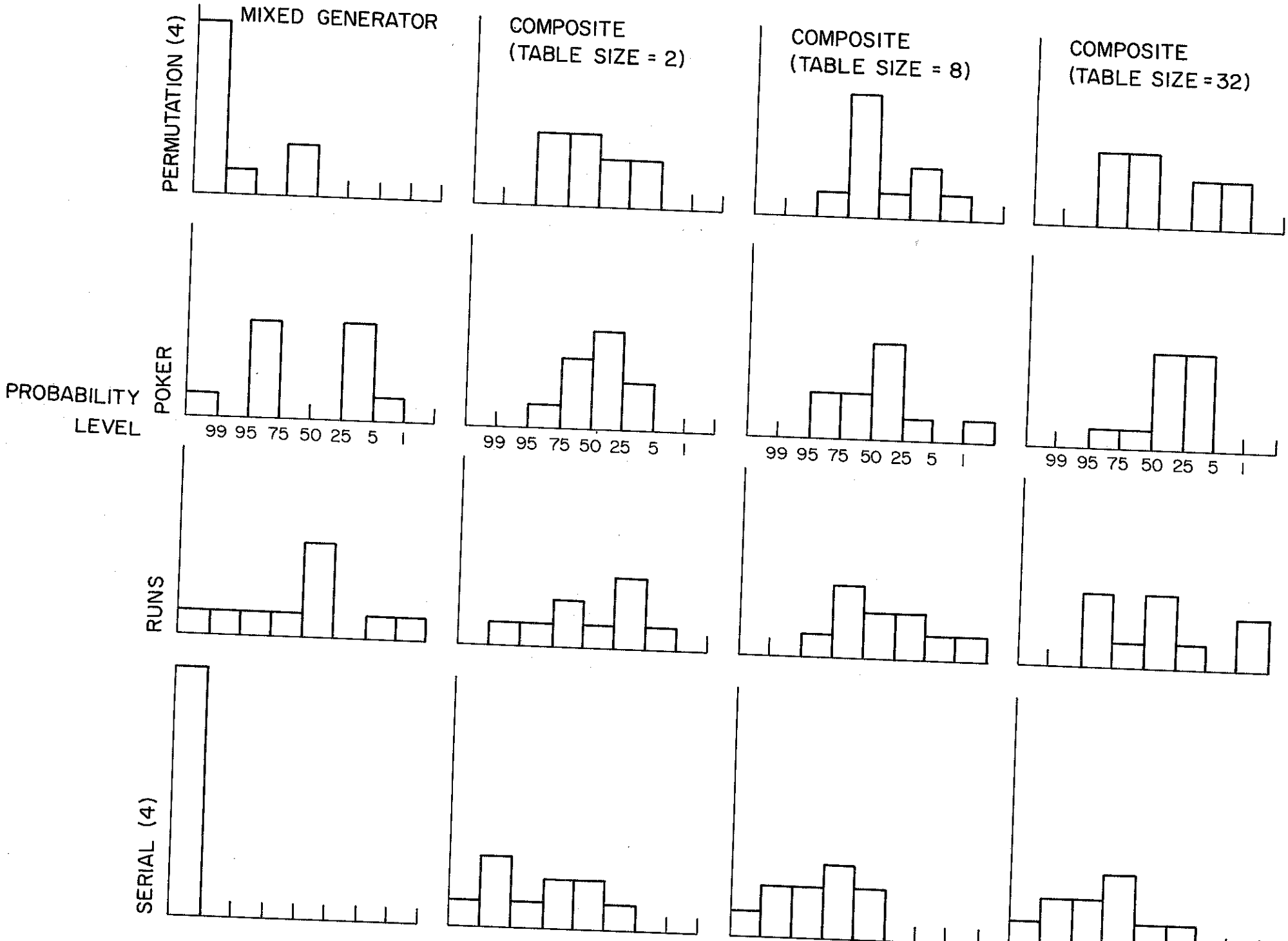

Figure 3. Continued.

is important simply because of the considerable savings in space and time. On reflection, the relatively good performance for a table size of 2 is not so unexpected. A simple counting argument reveals that, for this case, the possible number of subsequence orderings consisting of r members is 2^r. By the inherent properties of the generator, each ordering is equally likely. Consequently, a two element table effects a quite adequate shuffle.

To substantiate our findings, we replicated the table size experiment with the composite generators

$$Y_{n+1} \equiv (77 + 24J)\ Y_n + 371 \pmod{2^{11}} \qquad J = 0,1, \ldots, 9$$
$$Z_{n+1} \equiv 13Z_n \pmod{151}$$

Note that the indexing generator remains the same. A series of 10 sample sequences (actually subsequences), each of length 2000, were tested, and the results are shown in Figure 4. Again, the results indicate that shuffling effects considerable improvement. Without shuffling, 24 "bad" sequences are produced; while the composite generator produce no more than four. Comparing on the basis of "suspect" sequences, the primary generator alone produces 20, and the maximum produced by the composite generators is nine (for a table size of 2).

In total, using a table size of 2 seems to realize results comparable to 8 and 32. On the basis of these results, we see no indication that the higher overhead for a larger table is warranted.

CONCLUDING OBSERVATIONS

Limited to the information derived from the experiments described, we offer the following observations:

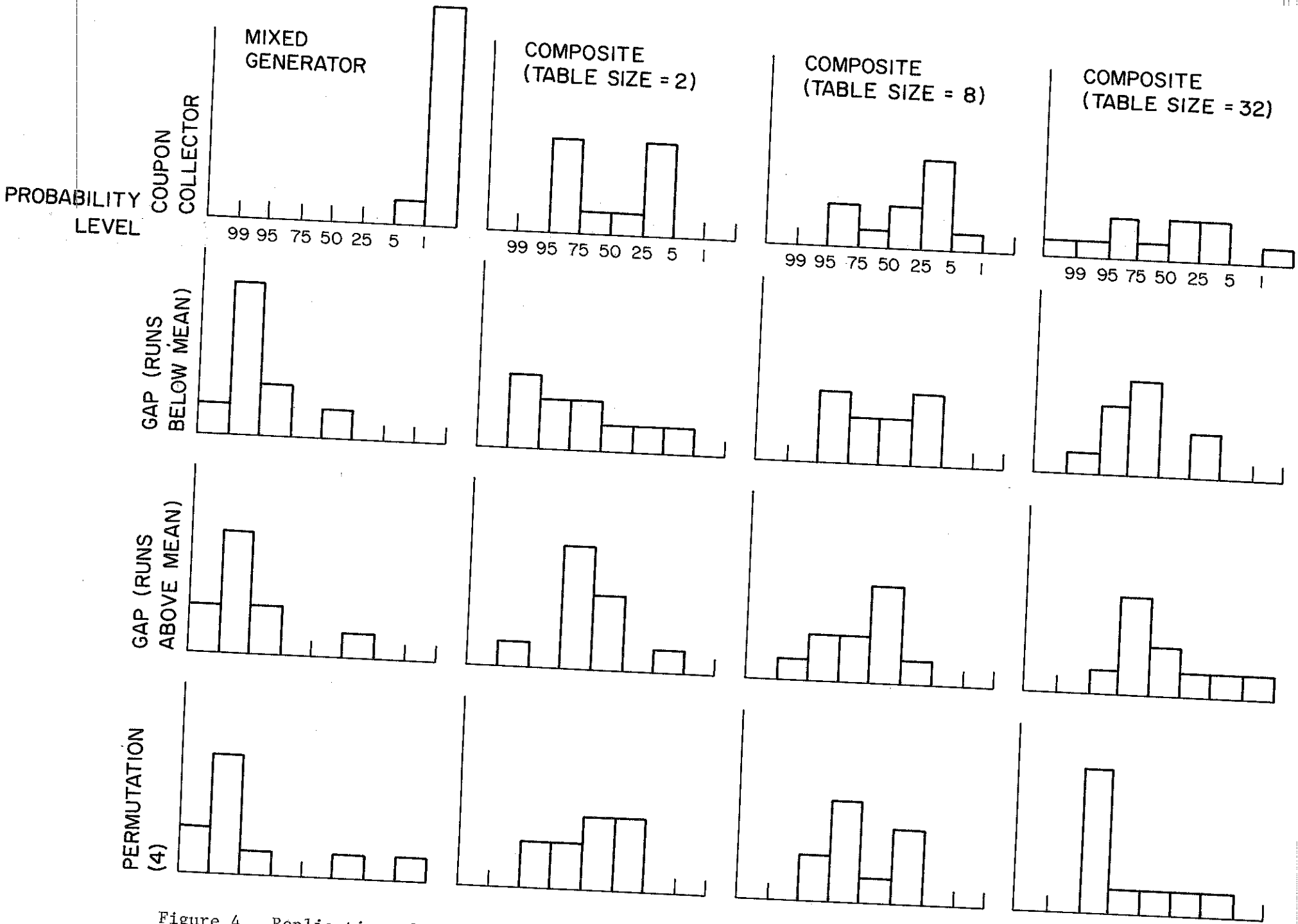

Figure 4. Replication of Table Size Effect on Statistical Behavior of Composite Generator.

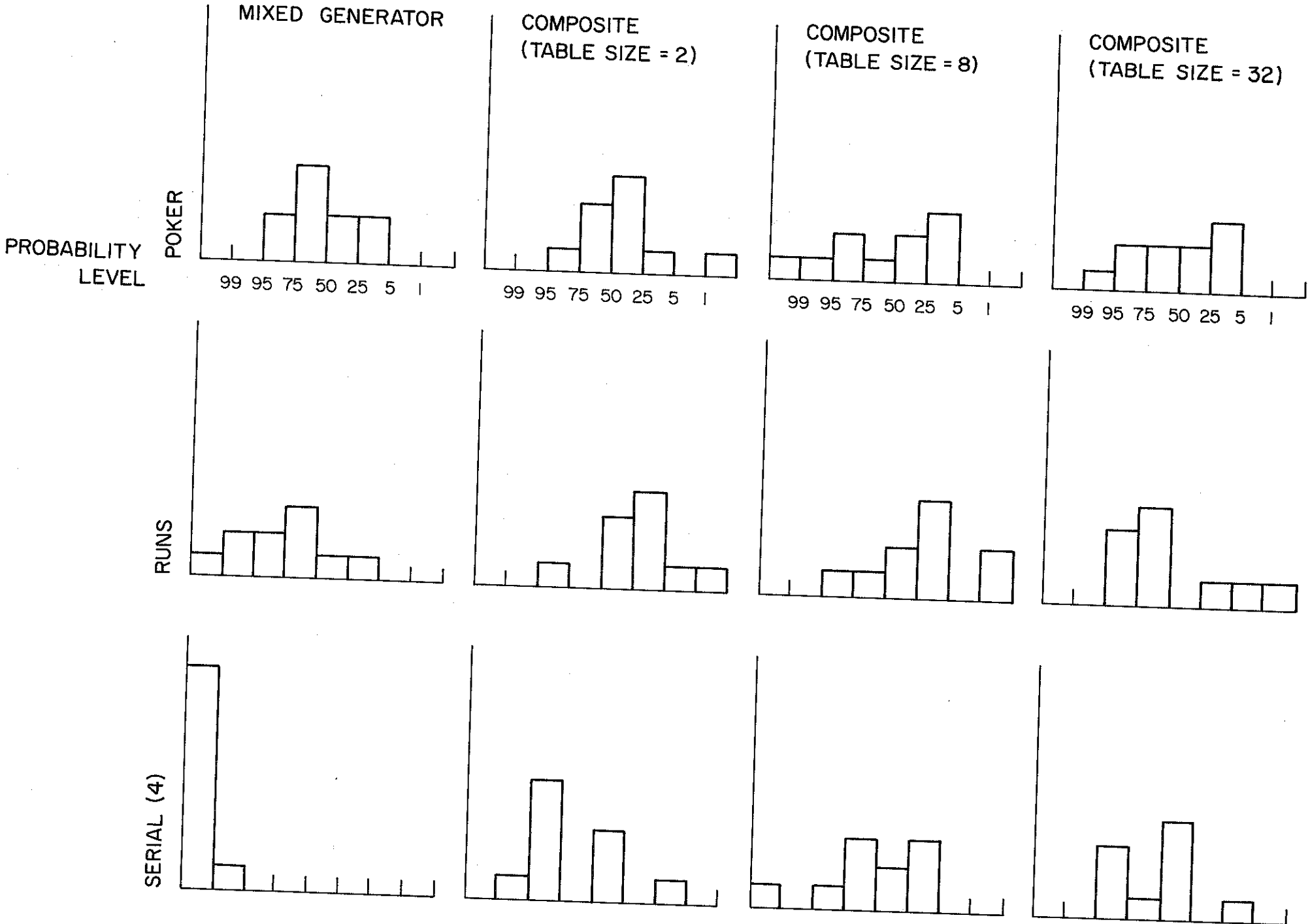

Figure 4. Continued.

(1) Shuffling accomplishes little on large word-length machines, where the modulus value invokes no serious limitation on the period. Simple generators, such as those above, provide comparable statistical behavior with less overhead.

(2) Shuffling can be used effectively to increase the period attainable on small word-length machines, where the modulus value represents a limitation on the period.

(3) To effect an increased period with a composite generator, the period of one generator should not be a divisor of the period of the other.

(4) A table size of 2 leads to results comparable with larger tables and incurs less overhead.

In the way of constructive comments on the third conclusion, let us suppose that a sample of no more than n random variates is needed. As a guide to constructing an acceptable composite generator, we propose the following:

(1) Let the primary generator be of the form

$$Y_{n+1} \equiv aY_n + c \pmod{2^b}$$

which has maximum period 2^b (a mixed generator).

(2) Choose the smallest prime modulus for an indexing generator of the form

$$Z_{n+1} \equiv \hat{a}\, Z_n \pmod{p}$$

having full period p-1 (a multiplicative generator),[4] such that

(a) $((p-1)/2,\ 2^b) = 1$, and

(b) the product of the prime factors of $(p-1)/2$ and 2^b exceeds $4n$.

This guideline assures that the sample includes no more than one-fourth the period of the composite generator.

---

[4]Recall that to obtain full period, $\hat{a}$ must be a primitive element mod p.

REFERENCES

1. Anderson, R. L. "Distribution of the Serial Correlation Coefficient," Annals of Mathematical Statistics, 13 (1): March 1942, 3-13.

2. Gebhardt, F. "Generating Pseudo-random Numbers by Shuffling a Fibonacci Sequence," Mathematics of Computation 21 (100): October 1967, 708-709.

3. Jansson, B. Random Number Generators, Peterson, Stockholm, 1966.

4. Knuth, D. E. The Art of Computer Programming: Volume 2/ Seminumerical Algorithms, Addison-Wesley, 1969.

5. Learmonth, G. P. and P. A. W. Lewis. "Statistical Tests of Some Widely Used and Recently Proposed Uniform Random Number Generators," NPS55LW73111A, Department of Operations Research and Administrative Sciences, Naval Postgraduate School, Monterey, California, November 1973.

6. Lehmer, D. H. "Mathematical Methods in Large-Scale Computing Units, Proceedings of the Second Symposium on Large-Scale Digital Computing Machinery, Cambridge, Harvard University Press, 1951, 141-145.

7. MacLaren, M. D. and G. Marsaglia. "Uniform Random Number Generators," Journal of the Association for Computing Machinery, 12 (1): January 1965, 83-89.

8. Marsaglia, G. "Random Numbers Fall Mainly in the Planes," Proceedings of the National Academy of Sciences, 61 (1): September 1968, 25-28.

9. Miller, J. C. and M. J. Prentice. "Additive Congruential Pseudo-random Number Generators," Computer Journal, 11 (3): November 1968, 341-346.

10. Nance, R. E. and C. L. Overstreet, Jr. "A Bibliography on Random Number Generation," Computing Reviews, 13 (10), October 1972, 495-508.

11. Nance, R. E. and C. L. Overstreet, Jr., "Implementation of FORTRAN Random Number Generators on Computers with Ones-Complement Arithmetic," Journal of Statistical Computation and Simulation, 4: 1975, 235-243.

12. Newman, T. G. and P. L. Odell. The Generation of Random Variates, Griffin's Statistical Monographs and Courses (No. 29), Hafner, 1971.

13. Overstreet, C. L., Jr. "A FORTRAN V Package for Testing and Analysis of Pseudorandom Number Generators," Technical Report CP-72009, CS/OR Center, Southern Methodist University, March 1972.

14. Page, E. S. "The Generation of Pseudo-Random Numbers," in S. H. Hollingsdale (ed.), Digital Simulation in Operational Research, American Elsevier Publishing Co.: New York, 1967, 55-63.

15. Van Gelder, A. "Some New Results in Pseudo-Random Number Generation," Journal of the Association for Computing Machinery, 14 (4): October 1967, 785-792.

16. Westlake, W. J. "A Uniform Random Number Generator Based on the Combination of Two Congruential Generators," Journal of the Association for Computing Machinery, 14 (2): April 1967, 337-340.