International Journal of Engineering & Technology
CENTRE OF PROFESSIONAL RESEARCH PUBLICATIONS
ISSN 2049-3444

International Journal of Engineering and Technology Volume 3 No. 4, April, 2013

Improved Design of Data-Oriented Universal Random Number Generator

Rasoul Farjami Nezhad, Reza Poorzare
Department of Computer Science, Islamic Azad University Tabriz Branch, Tabriz, Iran

ABSTRACT

In this paper a new design of universal data-oriented random number generator is introduced that improves previous design in front of consuming memory. Presented design will enable us to generate random number by modern computers with uniform, normal, exponential, and chi-square distributions in efficient and different way with improved memory consuming.

Keywords: *Probability tree; Random Number; Data–Oriented; Ahmad Fact; Digit Bank*

1. INTRODUCTION

Random numbers are useful for a variety of purposes, such as generating data encryption keys, simulating and modeling complex phenomena and for selecting samples from lager data sets.

Until now, due to lack of storage memory and low-speed, the problems were solved with much less data lying complex algorithms which are called function–oriented methods. Nowadays, growing memory technology facilitates computing system in managing large amount of data very fast and easily. Data oriented theory presents methods in which any concept can be modeled in terms of data structures. Thus, in modern computing system, Concepts that are modeled by data oriented theory recognizing and processing are very fast and questions can be answered by data processing or by fewer amounts of mathematical operations by using these models.

The methods which answer the questions by using large amount of data are called data–oriented methods.

This paper introduced an improved version of universal random number generator based on data oriented theory, that improves memory consuming and manage it to use optimal way. This model is compatible with modern computer's structure and it enables us to generate random number with four distributions and manage memory more than previous design. The rest of this paper organized as follows:

The related works of data oriented modeling and some basic concepts are presented in section two. The new design is explained in section three, and section four provides concluding remarks and future works.

2. RELATED WORKS

Data–oriented modeling is a useful and applied method. The basic structure of data–oriented modeling has been presented in [1]. New data–oriented modeling of uniform random number which is well-matched with computing systems has been presented in [2]. A novel method for improving the uniformity of random number generator named uniformity improving method, or UIM in short and data oriented model of uniform random variable named UDPD has been presented in [3]. A data oriented model of exponential random variable has been presented in [4] also a data–oriented model of normal random variable has been presented in [5] and chi– square random variable generator based on data oriented has been introduced in [6].

Each continuous random variable is a probabilistic model of non–deterministic phenomena that models it with pure mathematics, but when this variable is used in applied science, based on precision and importance of variable, its value represented by specific number of digits. For instance, depends on necessary, precision and importance random number can be used with one, two or more digits. In this paper all random numbers are generate with 3 digits. It can be important to have random number on special distributions. Integrated numbers with varied distributions in digit bank enable having accurate random numbers with necessary distribution.

Some introductory definitions such as probability digraph, digital prodigraph, value of walk, value of leaf, probability complete trees, n-complete probability trees and digital n-complete probability tree, have been defined in [1]. In this paper we use them, and some other definitions are provided as follows:

Definition:

T is probability complete tree if and only if the sum of the all adjacent weights of a vertex is either 1 or 0. Trees shown by Fig.1 and Fig.2 are probabilistically complete trees. Note that the vertices that have the total adjacency to edge weight of 0 are tree leaves [1].

Definition:

T is an n-complete probability tree if and only if:

1. It is a probabilistically complete tree.
2. Each of vertices that have total adjacency to edge weight of 0 must be in depth n of the tree root.

All leaves must be in depth n, which is the depth of tree. Fig.1 shows a 2-complete probability tree, but represented tree in Fig. 2 is not since it fails to satisfy the second part of this definition [1].

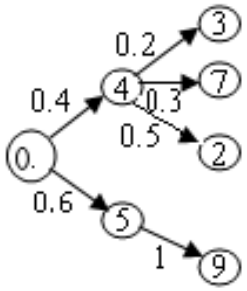

Fig. 1: A digital 2-complete probability tree

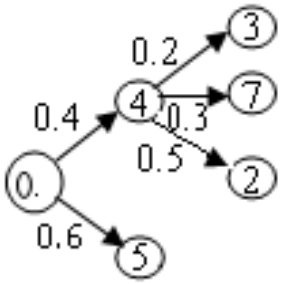

Fig. 2: A digital probabilistically complete tree

Definition:

Let t be a digital n-complete probability tree, and a_n be along with its leaf vertex on it. Suppose that equation 1:

$$Vol_{an}=y=0.a_1a_2\ldots a_n \quad (1)$$

Then P_y is the probability of vol_{an}, if and only if has equation 2:

$$P_y=P_{O.a1}*P_{a1a2}*\ldots*P_{an\text{-}1an}=P_{O.a1}\Pi_{i=2}P_{aiai+1} \quad (2)$$

Data-oriented model of random variable z is presented by calculating the probability of digits by AF[1] [11] and then these probabilities are used in probability complete tree as weights. AF is a fact that gives the probability of a random variable's digits by using its density function. In other words AF describes the digits of a random numbers that are randomly distributed and specify the probability of the kth digit be a given digit of a determined random number. Suppose that the random number z has probability density function f(z). $p_{z_1^i}$ denotes the probability of the first digit of z to be i where i=0,1,2,3,…,9 in decimal base. $p_{z_1^i}$ is calculated by using AF as equation 3:

$$p_{z_1^i}=p(i<z<i+1)=\int_i^{i+1} f(z)dz \quad (3)$$

As we know, this relation gives the probability that the value of random number z is in [i, i+1]. Now in view of AF, this is the probability that the first digit of z equals to i. For example $p_{z_1^2}$ shows the probability that first digit of z equals 2. This notation is used for making digital 1-probability complete tree. By calculating second digit's probability, digital 2-probability complete tree can be made. For this purpose $p_{z_2^{ij}}$ is used. $p_{z_2^{ij}}$, shows the probability that the second digit of z equals j if the first digit has occurred i. $p_{z_2^{ij}}$ is computed from equation 4:

$$p_{z_2^{ij}}=p(i.j<z<i.j+1)=\int_{i.j}^{i.j+1} f(z)dz \quad (4)$$

For example $p_{z_2^{34}}$ is the probability that z is in [3.4 , 3.5] and based on AF this is the probability that the second digit of z is 4 if the first digit has equaled to 3. For calculating kth digit probability $p_{z_k^{ij\ldots m}}$, is used where the probability of first, second,…, k-1th digits should be given.

For generating random numbers, data-oriented theory is used. In order to generate random numbers, some array like digits bank is considered. According to [1] for generating random numbers with two digits based on data-oriented theory, $p_{z_1^i}, p_{z_2^{ij}}$ are calculated for i,j=0 to 9 that results for $p_{z_1^i}$ are shown in table 1. Here 11 digits bank is considered and forms the value's obtained for $p_{z_1^i}$, the first two or three digits after decimal point are selected as the number of the iteration number for i and inserted in digits bank. For example if $p_{z_1^1}=0.228$, so that the number 22 is selected which is considered as the number of times that 1 is repeated and inserted in digits bank1. Table 2 shows the number times of a number in digit bank. Similarly three digits after decimal point of $p_{z_2^{ij}}$ are selected and inserted in digits bank2…11. After recording these numbers in the digits banks, and randomly shuffle (or exchange or rotate) the content of them, one number is selected randomly form digits bank1, then depending on the selected numbers, one of the other ten digits bank is selected and from this newly selected digits bank one element is chosen randomly, too. In this method, each digits bank has about 1000 elements with 4 bits for each

[1] Ahmad Fact

of them that are 11000 elements and 5.5kb memory using in total. The architecture of this case is shown in Fig.8.

Table. 1 Probability of fitrst digit

I	0	1	2	3	4	5	6	7	8	9
$p_{z_1^i}$	0.186	0.228	0.163	0.127	0.091	0.055	0.038	0.027	0.016	0.0103

Table. 2Number of digits in digits bank1

I	0	1	2	3	4	5	6	7	8	9
Number of digits in digits bank1	18	23	16	13	9	6	4	3	2	1

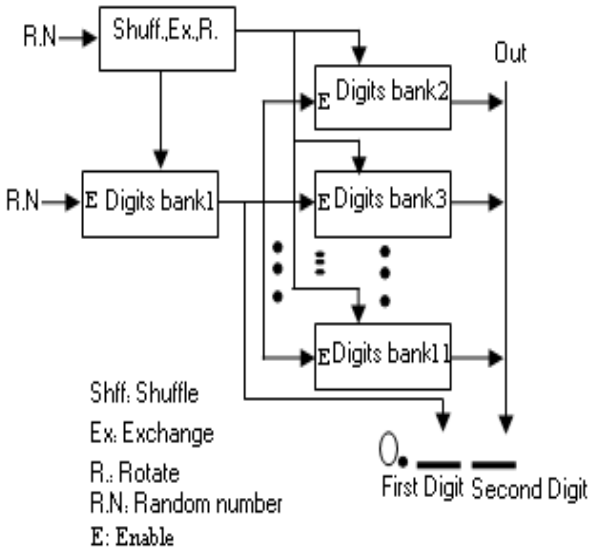

Fig. 3: Architecture of random number generator.

Simulation result shows that random number generator with two digits has more precision, but needs more memory space [1] and each of these models generates random numbers with any given distribution separately, but we suggested universal random numbers generator with less memory space. In this section U.R.N.G is designed as follows:

From the values obtained from $p_{z_2^{ij}}$ as shown in table3, the first three digits after decimal point is selected as the number of the iteration number for ij in numbers bank. In this paper instead of 11 digits banks, one number bank for each distribution is considered and by calculating $p_{z_2^{ij}}$ for all i , j of distributions and inserting selection results in number bank, UDRNG can start. At first, the type of distribution input by device and depending on the input distribution, one of the four numbers bank is selected, and by generating 2-bits digit randomly, content of selected numbers bank shuffle or exchange or rotate as shown in table4. Then, from selected number bank, one element is chosen randomly.

The hardware structure of URNG is shown in Fig4, to implement URNG in hardware structure we consider special memory or cache instate of numbers banks, decoder to select region of each distribution, 2-bit counter to select operation based on table3 to mix the content of cache.

Table 3: Number of digits in digits bank

Dist.	P\I	0	1	2	3	4	5	6	7	8	9
Uniform	$P_{u_1^i}$	$P_{u_1^0}$	$P_{u_1^1}$	$P_{u_1^2}$	$P_{u_1^3}$	$P_{u_1^4}$	$P_{u_1^5}$	$P_{u_1^6}$	$P_{u_1^7}$	$P_{u_1^8}$	$P_{u_1^9}$
Normal	$P_{N_1^i}$	$P_{N_1^0}$	$P_{N_1^1}$	$P_{N_1^2}$	$P_{N_1^3}$	$P_{N_1^4}$	$P_{N_1^5}$	$P_{N_1^6}$	$P_{N_1^7}$	$P_{N_1^8}$	$P_{N_1^9}$
Exponential	$P_{E_1^i}$	$P_{E_1^0}$	$P_{E_1^1}$	$P_{E_1^2}$	$P_{E_1^3}$	$P_{E_1^4}$	$P_{E_1^5}$	$P_{E_1^6}$	$P_{E_1^7}$	$P_{E_1^8}$	$P_{E_1^9}$
Chi-square	$P_{C_1^i}$	$P_{C_1^0}$	$P_{C_1^1}$	$P_{C_1^2}$	$P_{C_1^3}$	$P_{C_1^4}$	$P_{C_1^5}$	$P_{C_1^6}$	$P_{C_1^7}$	$P_{C_1^8}$	$P_{C_1^9}$

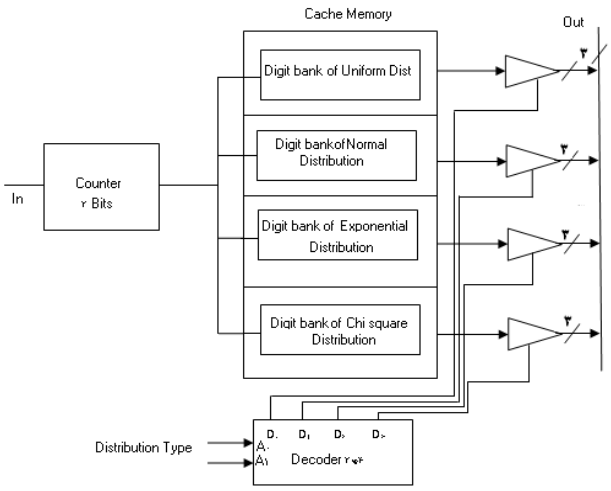

Fig.4: Hardware of universal random number generator

Now table 3 has some parameters in 2000 uniform numbers generated and 1000 numbers generated for normal, exponential, chi-square. Some parameters are shown in table 3 that the meaning of square error (MSE) for uniform is achieved from 5 and for other distributions achieved from 6.

Table 4: Instruction set

operation	Op-code
Shuffle	00
Rotate to right	01
Exchange	10
Rotate to left	11

$$MSE = \frac{1}{10}\sum_{i=0}^{9}\left(rf_i - \int_{100i}^{100(i+1)} f_s(s)ds\right)^2 \quad (5)$$

$$MSE = \frac{1}{100}\sum_{i=0}^{99}\left(rf_i - \int_{i}^{(i+1)} f_s(s)ds\right)^2 \quad (6)$$

where rf_i is the relative frequency of generated numbers in the interval [0.i,0.(i+1)] and

Table 5: Parameters table

Distribution	MSE	Memory cells used
Uniform	5.2e-5	2*1000*10bit=2.5kb
Normal	0.0253	2*1000*10bit=2.5kb
Exponential	0.0215	2*1000*10bit=2.5kb
Chi square	0.1219	2*1000*10bit=2.5kb

3. NEW DESIGN

In this section improved design of universal random number generator is introduced. To manage memory and decrease memory consuming, we save vol and pvol in records peer to peer like table 6,7 and content of tables transforms from array and placed in memory whatever it takes. In other words, generating random numbers in the new design is as follows:

1- One type of distribution is input by device.

2- Transform information of selected distribution to repetition unit and the number of occurrences of each digit in digits bank is written.

3- One of the operations from table 4 (shuffling, exchange, and shift) randomly selected and mixed numbers of digits bank.

4- The number is randomly selected from digit bank.

Table 6: Information of probability tree table

Walk	Value of Walk	Digit Bank
000	vow_0	n_0
001	vow_1	n_1
002	vow_2	n_2
⋮	⋮	
999	vow_{999}	n_{999}

Table 7: Information of probability tree table

Walk	Distribution Type							
	Uniform		Exponential		Normal		Chi-square	
	Vow	Digit Bank	Vow	Digit Bank	Vow	Digit Bank	Vow	Digit Bank
...	Vu۰	Nu۰	Ve۰	Ne۰	Vn۰	Nn۰	Vk۰	Nk۰
..۱	Vu۱	Nu۱	Ve۱	Ne۱	Vn۱	Nn۱	Vk۱	Nk۱
..۲	Vu۲	Nu۲	Ve۲	Ne۲	Vn۲	Nn۲	Vk۲	Nk۲
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
۹۹۹	Vu۹۹۹	Nu۹۹۹	Ve۹۹۹	Ne۹۹۹	Vn۹۹۹	Nn۹۹۹	Vk۹۹۹	Nk۹۹۹

By repeating this operation for many times (about 1000 times), form the recorded numbers, the cumulative frequency diagram can be obtained. This diagram matches the main diagram of distribution. Simulation is performed by MATLAB software and results are shown in Fig 6(a-d). This method can gives good and cumulative random

numbers with providence memory. The hardware structure is shown in Fig 5. This architecture is different from previous architecture (Fig4) that divided memory in four sections and we saw many numbers that iterate and lead to need and use lots of memory. Table 8 shows comparison of two architectures.

Table 8: Comparison of memory table

Distribution	Memory cells used In new design	Memory cells used In previous design
Uniform	2.5kb	2.5kb
Normal	2.5kb	2.5kb
Exponential	2.5kb	2.5kb
Chi square	2.5kb	2.5kb
Sum	10	10kb

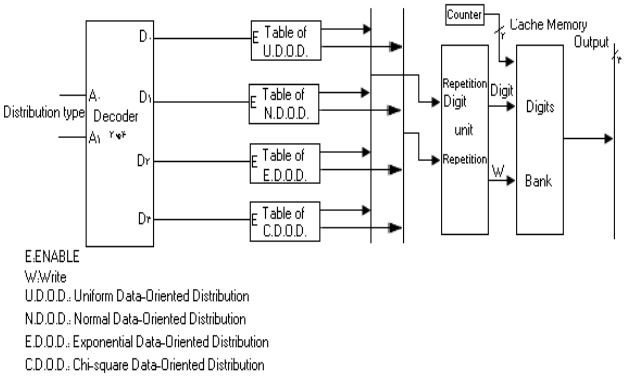

Fig. 5: Hardware of new model of universal random number generator

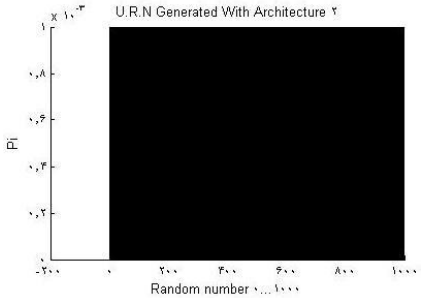

Fig.6.(a): UDRNG for uniform distribution

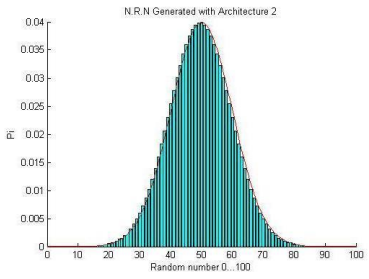

Fig.6.(b) UDRNG for normal distribution

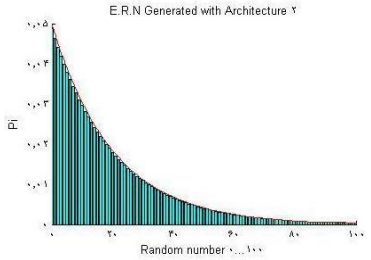

Fig.6.(c) UDRNG for exponential distribution

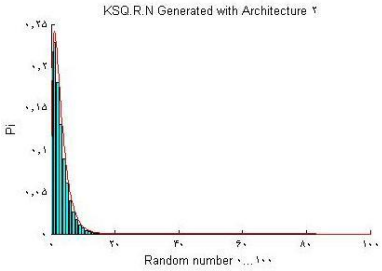

Fig.6.(d) UDRNG for chi-square distribution

4. CONCLUSION AND FURTHER WORKS

In this paper we presented an improved universal random number generator based on data oriented by decreasing memory consumption in comparision of previous design, Because of integration and this concept that for all depth of probability tree we can suppose it in one depth, Results shows that new design caused to have a 12.5% reduction in memory (table8).We believe that still by new techniques we can decrease memory consumption and execution time too, we expect data-oriented model to be used by computer to do probability and statistical inference efficiently. To access high performance, we try to use some efficient methods for increasing the speed of this approach and to achieve this goal, we will design some special memories that match this method, and shuffling can be done in it very quickly.

REFERENCES

[1] Habibzad-navin,A. ,E.S. Alikhani, M.Mirnia and S.Y. Torabi, "chi squre Random Variable Generator" India.2010 computational Intelligence and communication net works, PP.460-465.

[2] Habibzad-navin, A. and M.Mirnia, "Alternative views of the shortest path problem" in proceeding of the international Mathematical conference, Iran, 1999, PP.122-124.

[3] Habibizad-navin, A. and M.naghian Fesharaki and M.Teshnelab and M.Mirnia, "Probability graph and

some of ios important structure". In proceeding of the 5th Seminar on probability and Statistic processes. Birjand, Iran, 2005, PP. 155-160.

[4] Habibizad-navin, A. and M.naghian Fesharaki and M.Teshnelab and M.Mirnia, "data – oriented modeling of uniform random variable: applied approach" In proceeding of World Academy of science, Engineering and Technology. Vienna, Austria, 2007, PP.382-385.

[5] Olfatkhah, R. and Habibzad-navin, A. and M.Mirnia, "Anovel model of normal random variable based on data –oriented theory", ICCEA. International conference on computer Engineering and applications, in press, 2010.

[6] Habibizad–navin , A. and R.Olfatkhah and M.Mirnia, "A Data – oriented model of exponential Random Varible", ICACC. International conference of Advanced computer control china,2010, PP.603-607.

