Unbiased Random Sequences from Quasigroup String Transformations

Smile Markovski[1], Danilo Gligoroski[1], and Ljupco Kocarev[2]

[1] "Ss Cyril and Methodius" University
Faculty of Natural Sciences and Mathematics, Institute of Informatics
P.O.Box 162, 1000 Skopje, Republic of Macedonia
{danilo,smile}@ii.edu.mk
[2] University of California San Diego
Institute for Nonlinear Science
9500 Gilman Drive, La Jolla, CA 92093-0402, USA
lkocarev@ucsd.edu

Abstract. The need of true random number generators for many purposes (ranging from applications in cryptography and stochastic simulation, to search heuristics and game playing) is increasing every day. Many sources of randomness possess the property of stationarity. However, while a biased die may be a good source of entropy, many applications require input in the form of unbiased bits, rather than biased ones. In this paper, we present a new technique for simulating fair coin flips using a biased, stationary source of randomness. Moreover, the same technique can also be used to improve some of the properties of pseudo random number generators. In particular, an improved pseudo random number generator has almost unmeasurable period, uniform distribution of the letters, pairs of letters, triples of letters, and so on, and passes many statistical tests of randomness. Our algorithm for simulating fair coin flips using a biased, stationary source of randomness (or for improving the properties of pseudo random number generators) is designed by using quasigroup string transformations and its properties are mathematically provable. It is very flexible, the input/output strings can be of 2-bits letters, 4-bits letters, bytes, 2-bytes letters, and so on. It is of linear complexity and it needs less than 1Kb memory space in its 2-bits and 4-bits implementations, hence it is suitable for embedded systems as well.

1 Introduction

Random number generators (RNGs) are useful in every scientific area which uses Monte Carlo methods. It is difficult to imagine a scientific area where Monte Carlo methods and RNGs are not used. Extremely important is the application of RNGs in cryptography for generation of cryptographic keys, and random initialization of certain variables in cryptographic protocols. Countless applications in cryptography, stochastic simulation, search heuristics, and game playing rely on the use of sequences of random numbers.

The choice of the RNG for a specific application depends on the requirements specific to the given application. If the ability to regenerate the random sequence is of crucial significance such as debugging simulations, or the randomness requirements are not very stringent (flying through space on your screen saver), or the hardware generation costs are unjustified, then one should resort to pseudo-random number generators (PRNGs). PRNGs are algorithms implemented on finite-state machines and are capable of generating sequences of numbers which appear random-like from many aspects. Though they are necessarily periodic ("Anyone who considers arithmetical methods of producing random digits is, of course, in a state of sin", John von Neumann), their periods are very long, they pass many statistical tests and can be easily implemented with simple and fast software routines.

It is widely accepted that the core of any RNG must be an intrinsically random physical process. So, it is no surprise that the proposals and implementations of RNGs range from tossing a coin, throwing a dice, drawing from a urn, drawing from a deck of cards and spinning a roulette to measuring thermal noise from a resistor and shot noise from a Zener diode or a vacuum tube, measuring radioactive decay from a radioactive source, integrating dark current from a metal insulator semiconductor capacitor, detecting locations of photoevents, and sampling a stable high-frequency oscillator with an unstable low-frequency clock. Some of the sources of randomness, such as radioactive sources [1] and quantum-mechanical sources [2], may yield data from probability distributions that are stationary. Therefore, the output of these sources does not change over time and does not depend on previous outputs. However, even if a source is stationary, it generally has a bias. In other words, the source does not give unbiased bits as direct output. It is therefore quite important to be able to extract unbiased bits efficiently from a stationary source with unknown bias.

Suppose that a reading obtained from a stationary source of randomness can be equal to any one of m different values, but that the probability of obtaining any one of these values is unknown and in general not equal to $1/m$. In other words, we assume that the source may be loaded. Our aim in this paper is to simulate unbiased coin flips using a biased source.

Previous Work – There are several references to the problem of simulating unbiased physical sources of randomness. Von Neumann [3] described the following method; flip the biased coin twice: if it comes up HT, output an H, if it comes up TH, output a T, otherwise, start over. This method will simulate the output of an unbiased coin irrespective of the bias of the coin used in the simulation. Elias [4] proposed a method of extracting unbiased bits from biased Markov chains. Stout and Warren [5] and Juels et. al. [6] presented new extensions of the technique of von Neumann. Stout and Warren suggested a method for simulating a fixed number of fair coin flips using as few rolls of a biased die as possible, while the authors of [6] proposed an algorithm for extracting, given a fixed number of rolls of a biased die, as many fair coin flips as possible. The general characteristics of the methods for simulating unbiased physical sources of randomness are: (i) all of them do not use each bit of information generated by the source, (ii) some of the methods can be implemented in computationally effective way, but for some of them corresponding algorithms are of exponential nature and then approximations should be involved, and (iii) for some of them mathematical proofs are supplied for their properties.

Our Work – In this paper we propose a method for simulating unbiased physical sources of randomness which is based on the quasigroup string transformations and some of their provable properties. Our method uses each bit of information produced by a discrete source of randomness. Moreover, our method is capable of producing a random number sequence from a very biased stationary source (for example, from a source that produces 0 with probability 1/1000 and 1 with probability 999/1000). The complexity of our algorithm is linear, i.e. an output string of length n will be produced from an input string of length n with complexity $O(n)$. Our algorithm is highly parallel. This means there exist computationally very effective software and hardware implementations of the method. Our algorithm is also very flexible: the same design can be used for strings whose letters consists of 2-bits, 4-bits, bytes, 2-bytes, and generally it can be designed for an arbitrary n-bit letters alphabet ($n \geq 2$). The method proposed in this paper can also be used to improve the quality of existing PRNGs so that they pass many statistical tests and their periods can be arbitrary large numbers. Since many of the weak PRNGs are still in use because of the simplicity of their design and the speed of producing pseudo random strings (although of bad quality), our method in fact can improve the quality of these PRNGs very effectively.

The paper is organized as follows. Needed definitions and properties of quasigroups and quasigroup string transformations are given in Section 2. The algorithm for simulating unbiased physical sources of randomness (or for improving PRNGs) is presented in Section 3. In this section we also present some numerical results concerning our method, while the proofs of the main theorems are given in the appendicitis. In Section 4 we close our paper with conclusion.

2 Quasigroup string transformations

Here we give a brief overview of quasigroups, quasigroup operations and quasigroup string transformations (more detailed explanation the reader can find in [7], [8]).

A quasigroup is a groupoid $(Q, *)$ satisfying the laws

$$(\forall u, v \in Q)(\exists x, y \in Q)(u * x = v,\ y * u = v),$$

$$x * y = x * z \implies y = z,\ y * x = z * x \implies y = z.$$

Hence, a quasigroup satisfies the cancelation laws and the equations $a * x = b,\ y * a = b$ have unique solutions $x,\ y$ for each $a, b \in Q$. If $(Q, *)$ is a quasigroup, then $*$ is called a quasigroup operation.

Here we consider only finite quasigroups, i.e Q is a finite set. Closely related combinatorial structures to finite quasigroups are the so called Latin squares: a

Latin square L on a finite set Q (with cardinality $|Q| = s$) is an $s \times s$-matrix with elements from Q such that each row and each column of the matrix is a permutation of Q. To any finite quasigroup $(Q, *)$ given by its multiplication table it is associated a Latin square L, consisting of the matrix formed by the main body of the table, and each Latin square L on a set Q define a quasigroup $(Q, *)$.

Given a quasigroup $(Q, *)$ five new operations, so called parastrophes or adjoint operations, can be derived from the operation $*$. We will need only the following two, denoted by $\backslash$ and $/$, and defined by:

$$x * y = z \iff y = x \backslash z \iff x = z/y \tag{1}$$

Then the algebra $(Q, *, \backslash, /)$ satisfies the identities

$$x \backslash (x * y) = y,\ x * (x \backslash y) = y,\ (x * y)/y = x,\ (x/y) * y = x \tag{2}$$

and $(Q, \backslash)$, $(Q, /)$ are quasigroups too.

Several quasigroup string transformations can be defined and those of interest of us will be explained bellow. Consider an alphabet (i.e. a finite set) A, and denote by A^+ the set of all nonempty words (i.e. finite strings) formed by the elements of A. The elements of A^+ will be denoted by $a_1 a_2 \dots a_n$ rather than $(a_1, a_2, \dots, a_n)$, where $a_i \in A$. Let $*$ be a quasigroup operation on the set A. For each $l \in A$ we define two functions $e_{l,*},\ e'_{l,*} : A^+ \longrightarrow A^+$ as follows. Let $a_i \in A,\ \alpha = a_1 a_2 \dots a_n$. Then

$$e_{l,*}(\alpha) = b_1 \dots b_n \iff b_{i+1} = b_i * a_{i+1} \tag{3}$$

$$e'_{l,*}(\alpha) = b_1 \dots b_n \iff b_{i+1} = a_{i+1} * b_i \tag{4}$$

for each $i = 0, 1, \dots, n-1$, where $b_0 = l$. The functions $e_{l,*}$ and $e'_{l,*}$ are called e- and e'-transformations of A^+ based on the operation $*$ with leader l. Graphical representations of the e- and e'-transformations are shown on Figure 1 and Figure 2.

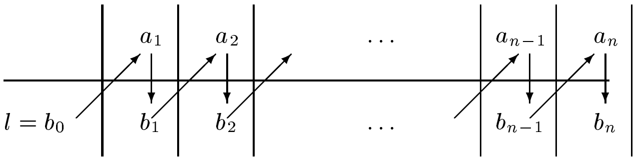

Fig. 1. Graphical representation of an e-transformation

Example 1. Take $A = \{0, 1, 2, 3\}$ and let the quasigroup $(A, *)$ be given by the multiplication scheme

$*$	0	1	2	3
0	2	1	0	3
1	3	0	1	2
2	1	2	3	0
3	0	3	2	1

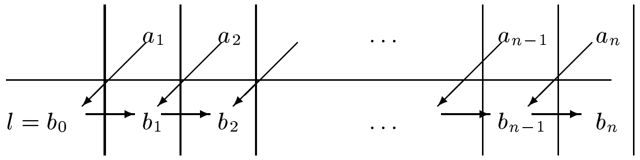

Fig. 2. Graphical representation of an e'-transformation

Consider the string $\alpha = 1\,0\,2\,1\,0\,0\,0\,0\,0\,0\,0\,0\,0\,1\,1\,2\,1\,0\,2\,2\,0\,1\,0\,1\,0\,3\,0\,0$ and choose the leader 0. Then we have the following transformed strings
$e_{0,*}(\alpha) = 1\,3\,2\,2\,1\,3\,0\,2\,1\,3\,0\,2\,1\,0\,1\,1\,2\,1\,1\,1\,3\,3\,0\,1\,3\,1\,3\,0,$
$e'_{0,*}(\alpha) = 3\,3\,0\,3\,3\,3\,3\,3\,3\,3\,3\,3\,3\,2\,1\,2\,1\,1\,2\,3\,3\,2\,0\,3\,3\,1\,1\,1.$

Four consecutive applications of these transformations are presented below:

$$\begin{array}{c|l}
 & 1\,0\,2\,1\,0\,0\,0\,0\,0\,0\,0\,0\,0\,1\,1\,2\,1\,0\,2\,2\,0\,1\,0\,1\,0\,3\,0\,0 = \alpha \\
\hline
0 & 1\,3\,2\,2\,1\,3\,0\,2\,1\,3\,0\,2\,1\,0\,1\,1\,2\,1\,1\,1\,3\,3\,0\,1\,3\,1\,3\,0 = e_{0,*}(\alpha) \\
0 & 1\,2\,3\,2\,2\,0\,2\,3\,3\,1\,3\,2\,2\,1\,0\,1\,1\,2\,2\,2\,0\,3\,0\,1\,2\,2\,0\,2 = e_{0,*}{}^2(\alpha) \\
0 & 1\,1\,2\,3\,2\,1\,1\,2\,0\,1\,2\,3\,2\,2\,1\,0\,1\,1\,1\,1\,3\,1\,3\,3\,2\,3\,0\,0 = e_{0,*}{}^3(\alpha) \\
0 & 1\,0\,0\,3\,2\,2\,2\,3\,0\,1\,1\,2\,3\,2\,2\,1\,0\,1\,0\,1\,2\,2\,0\,3\,2\,0\,2\,1 = e_{0,*}{}^4(\alpha)
\end{array}$$

$$\begin{array}{c|l}
 & 1\,0\,2\,1\,0\,0\,0\,0\,0\,0\,0\,0\,0\,1\,1\,2\,1\,0\,2\,2\,0\,1\,0\,1\,0\,3\,0\,0 = \alpha \\
\hline
0 & 3\,3\,0\,3\,3\,3\,3\,3\,3\,3\,3\,3\,3\,2\,1\,2\,1\,1\,2\,3\,3\,2\,0\,3\,3\,1\,1\,1 = e'_{0,*}(\alpha) \\
0 & 0\,0\,2\,2\,2\,2\,2\,2\,2\,2\,2\,2\,2\,3\,2\,3\,2\,1\,2\,2\,2\,3\,3\,1\,3\,2\,1\,0 = e'_{0,*}{}^2(\alpha) \\
0 & 2\,0\,1\,2\,3\,0\,1\,2\,3\,0\,1\,2\,3\,1\,2\,2\,3\,2\,3\,0\,1\,3\,1\,0\,0\,1\,0\,2 = e'_{0,*}{}^3(\alpha) \\
0 & 1\,1\,0\,1\,3\,3\,2\,3\,1\,1\,0\,1\,3\,2\,3\,0\,0\,1\,3\,3\,2\,2\,1\,1\,1\,0\,2\,3 = e'_{0,*}{}^4(\alpha)
\end{array}$$

One can notice that the starting distribution of 0, 1, 2 and 3 in α : 16/28, 7/28, 4/28, 1/28 is changed to 7/28, 7/28, 10/28, 4/28 in $e_{0,*}{}^4(\alpha)$ and to 5/28, 10/28, 5/28, 8/28 in $e'_{0,*}{}^4(\alpha)$, hence the distributions became more uniform.

Several quasigroup operations can be defined on the set A and let $*_1$, $*_2$, $\dots$, $*_k$ be a sequence of (not necessarily distinct) such operations. We choose also leaders $l_1,\ l_2,\ \dots,\ l_k \in A$ (not necessarily distinct either), and then the compositions of mappings

$$E_k = E_{l_1 \dots l_k} = e_{l_1,*_1} \circ e_{l_2,*_2} \circ \cdots \circ e_{l_k,*_k},$$

$$E'_k = E'_{l_1 \dots l_k} = e'_{l_1,*_1} \circ e'_{l_2,*_2} \circ \cdots \circ e_{l_k,*_k},$$

are said to be E- and E'-transformations of A^+ respectively. The functions E_k and E'_k have many interesting properties, and for our purposes the most important ones are the following:

Theorem 1. ([8]) *The transformations E_k and E'_k are permutations of A^+.*

Theorem 2. ([8]) *Consider an arbitrary string $\alpha = a_1 a_2 \dots a_n \in A^+$, where $a_i \in A$, and let $\beta = E_k(\alpha)$, $\beta' = E'_k(\alpha)$. If n is large enough integer then, for each l : $1 \le l \le k$, the distribution of substrings of β and β' of length l is uniform. (We note that for $l > k$ the distribution of substrings of β and β' of length l may not be uniform.)*

We say that a string $\alpha = a_1 a_2 \dots a_n \in A^+$, where $a_i \in A$, has a period p if p is the smallest positive integer such that $a_{i+1}a_{i+2}\dots a_{i+p} = a_{i+p+1}a_{i+p+2}\dots$ $\dots a_{i+2p}$ for each $i \geq 0$. The following property holds:

Theorem 3. ([9]) *Let $\alpha = a_1 a_2 \dots a_n \in A^+$, $a_i \in A$, and let $\beta = E_k(\alpha)$, $\beta' = E'_k(\alpha)$, where $E_k = E_{aa\dots a}$, $E'_k = E'_{aa\dots a}$, $a \in A$ and $a * a \neq a$. Then the periods of the strings β and β' are increasing at least linearly by k.*

We should note that the increasing of the periods depends of the number of quasigroup transformations k, and for some of them it is exponential, i.e. if α has a period p, then $\beta = E_k(\alpha)$ and $\beta' = E'_k(\alpha)$ may have periods greater than $p\,2^{c\,k}$, where c is some constant. We will discuss this in more details in the next section. In what follows we will usually use only E-transformations, since the results will hold for E'-transformations by symmetry.

Theorem 1 is easy to prove (and one can find the proof in [8]). The proofs of Theorem 2 and Theorem 3 are given in the Appendix I and the Appendix II, respectively.

3 Description of the algorithms

Assume that we have a discrete biased stationary source of randomness which produces strings from A^+, i.e. the alphabet of source is A, where

$$A = \{a_0, a_1, \dots, a_{s-1}\}$$

is a finite alphabet. (However, we may also think that strings in A^+ are produced by a PRNG.)

Now we define two algorithms for simulating unbiased physical sources of randomness (or for improving PRNGs), based on E- and E'-transformations accordingly. We call them an $E-algorithm$ and an $E'-algorithm$. In these algorithms we use several internal and temporal variables b, L_1, $\dots$, L_n. The input of the algorithm is the order of the quasigroup s, a quasigroup $(A, *)$ of order s, a fixed element $l \in A$ (the leader), an integer k giving the number of applications of the transformations $e_{l,*}$ and $e'_{l,*}$ and a biased random string $b_0, b_1, b_2, b_3, \dots$. The output is an unbiased random string.

The performance of the algorithms is based on Theorems 1, 2 and 3. By Theorem 1 we have that $E-algorithm$ and $E'-algorithm$ are injective, meaning that different input string produces different output string. Theorem 2 guarantees that the algorithms generate unbiased output random strings. Theorem 3 guarantees that if the biased source has period p (such as some Pseudo Random Number Generator) the algorithm will generate unbiased output with longer period.

Both $E-algorithm$ and $E'-algorithm$ can also be used to improve the properties of PRNGs. For example, for suitable choice of the quasigroup and suitable choice of the parameter s, Theorem 3 shows that the period of the output pseudo random string can be made arbitrary large. In addition, we have checked the quality of output pseudo random strings by using available statistical tests (such as Diehard [10] and those suggested by NIST [11]) for different quasigroups, leaders, and different values of n: in all these cases the pseudo strings passed all of the tests.

E-algorithm
Phase I. Initialization
1. Choose a positive integer $s \geq 4$;
2. Choose a quasigroup $(A, *)$ of order s;
3. Set a positive integer k;
4. Set a leader l, a fixed element of A such that $l * l \neq l$;
Phase II. Transformations of the random string $b_0 b_1 b_2 b_3 \ldots, \quad b_j \in A$
5. For $i = 1$ to k do $L_i \leftarrow l$;
6. $j \leftarrow 0$;
7. do
$b \leftarrow b_j$;
$L_1 \leftarrow L_1 * b$;
For $i = 2$ to k do $L_i \leftarrow L_i * L_{i-1}$;
Output: L_k;
$j \leftarrow j + 1$;
loop;

The $E' - algorithm$ differs of the $E - algorithm$ only in step 7:

$E' - algorithm$
$7'$. do
$b \leftarrow b_j$;
$L_1 \leftarrow b * L_1$;
For $i = 2$ to k do $L_i \leftarrow L_{i-1} * L_i$;
Output: L_k;
$j \leftarrow j + 1$;
loop;

Example 2. The PRNG used in GNU C v2.03 do not passed all of the statistical tests in the Diehard Battery v0.2 beta [10], but the improved PRNG passed all of them after only one application ($k = 1$) of an e-transformation performed by a quasigroup of order 256. The results are given in the next two screen dumps.

```
***** TEST SUMMARY FOR GNU C (v2.03) PRNG *****
All p-values:
0.2929,0.8731,0.9113,0.8755,0.4637,0.5503,0.9435,0.7618,0.9990,0.0106,1.0000,0.0430,0.0680,
1.0000,1.0000,1.0000,1.0000,1.0000,1.0000,1.0000,1.0000,1.0000,1.0000,1.0000,1.0000,1.0000,
1.0000,1.0000,1.0000,0.0000,1.0000,1.0000,1.0000,1.0000,1.0000,1.0000,1.0000,1.0000,1.0000,
1.0000,1.0000,1.0000,1.0000,1.0000,1.0000,1.0000,0.2009,0.0949,0.1939,0.0944,0.2514,0.3419,
0.5714,0.2256,0.1484,0.7394,0.0562,0.3314,0.2559,0.5677,0.3061,0.4763,0.8185,0.1571,0.2072,
0.5667,0.7800,0.6428,0.7636,0.1529,0.9541,0.8689,0.1558,0.6235,0.5275,0.6316,0.7697,0.7181,
```

0.7921,0.4110,0.3050,0.8859,0.4783,0.3283,0.4073,0.2646,0.0929,0.6029,0.4634,0.8462,0.3777,
0.2385,0.6137,0.1815,0.4001,0.1116,0.2328,0.0544,0.4320,0.0000,0.0000,0.0000,0.0000,0.0000,
0.0000,0.0000,0.0000,0.0000,0.0000,0.0000,0.0000,0.0000,0.0000,0.0000,0.0000,0.0000,0.0000,
0.0000,0.0000,0.0003,0.0000,0.0000,0.0000,0.0000,0.0000,0.0000,1.0000,0.0013,0.0000,0.0000,
0.0000,0.0000,0.0000,0.0000,1.0000,0.0000,0.0000,0.0000,0.0000,0.0000,0.0000,0.0000,1.0000,
0.0003,0.0000,0.0000,0.0000,0.0000,0.0000,0.0000,0.0000,0.0000,0.0000,0.0000,0.0000,0.0000,
0.0753,0.0010,0.0000,0.0000,0.0000,0.0000,0.0000,0.0000,0.0233,0.0585,0.0000,0.0000,0.0000,
0.0000,0.0000,0.0000,0.2195,0.0321,0.0000,0.0000,0.9948,0.0006,0.0000,0.0000,0.0688,0.5102,
0.6649,0.1254,0.2967,0.1218,0.8199,0.7125,0.6873,0.1663,0.7150,0.7275,0.9035,0.1946,0.7261,
0.7243,0.1083,0.4266,0.7664,0.8384,0.7317,0.8340,0.3155,0.0987,0.7286,0.6645,0.9121,0.0550,
0.6923,0.1928,0.7236,0.0159,0.4636,0.2764,0.2325,0.3406,0.3746,0.1208,0.8145,0.3693,0.7426,
0.6272,0.6139,0.4957,0.3623,0.4929,0.3628,0.5266,0.2252,0.7948,0.7327,0.2732,0.6895,0.2325,
0.2303,0.1190,0.8802,0.0377,0.6887,0.4175,0.0803,0.3687,0.7010,0.7425,0.1003,0.0400,0.5055,
0.9488,0.3209,0.5965,0.0676,0.0021,0.2337,0.5204,0.5343,0.0630,0.2008,0.6496,0.4157,0.0559,
0.9746,0.1388,0.4657,0.5793,0.6455,0.8441,0.5248,0.7962,0.8870

Overall p-value after applying KStest on 269 p-values = 0.000000

*** TEST SUMMARY FOR GNU C v2.03 + QUASIGROUP PRNG IMPROVER ***

All p-values:

0.5804,0.3010,0.1509,0.5027,0.3103,0.5479,0.3730,0.9342,0.4373,0.5079,0.0089,0.3715,0.3221,
0.0584,0.1884,0.1148,0.0662,0.8664,0.5070,0.7752,0.1939,0.9568,0.4948,0.1114,0.2042,0.4190,
0.4883,0.4537,0.0281,0.0503,0.0346,0.6085,0.1596,0.1545,0.0855,0.5665,0.0941,0.7693,0.0288,
0.1372,0.8399,0.0320,0.6930,0.3440,0.9842,0.9975,0.1354,0.8776,0.1919,0.2584,0.6437,0.1995,
0.2095,0.3298,0.5180,0.8136,0.7294,0.7560,0.0458,0.6285,0.1775,0.1546,0.0397,0.5135,0.0938,
0.6544,0.9673,0.8787,0.9520,0.8339,0.4397,0.3687,0.0044,0.7146,0.9782,0.7440,0.3042,0.3388,
0.8465,0.7123,0.8752,0.8775,0.7552,0.5711,0.3768,0.1390,0.9870,0.9444,0.6101,0.1090,0.2032,
0.8538,0.6871,0.8785,0.9159,0.4128,0.4513,0.1512,0.8808,0.7079,0.2278,0.1400,0.6461,0.4082,
0.3353,0.1064,0.6739,0.2066,0.5119,0.0558,0.5748,0.5064,0.8982,0.6422,0.7512,0.8633,0.1712,
0.4625,0.0843,0.0903,0.7641,0.6253,0.8523,0.7768,0.8041,0.5360,0.0826,0.0378,0.8710,0.4901,
0.7994,0.7748,0.8403,0.9886,0.1373,0.7082,0.8860,0.9595,0.2671,0.0038,0.7572,0.8403,0.7410,
0.5615,0.6181,0.1257,0.5960,0.2432,0.8302,0.1981,0.7764,0.2109,0.2109,0.6620,0.8938,0.0052,
0.8116,0.5196,0.0836,0.4144,0.2466,0.3298,0.8724,0.9837,0.8748,0.0930,0.5055,0.6511,0.3569,
0.2832,0.4029,0.9290,0.3470,0.6598,0.4796,0.3758,0.6077,0.4213,0.1886,0.1500,0.3341,0.0594,
0.0663,0.0946,0.8279,0.2451,0.2969,0.9297,0.0739,0.4839,0.1307,0.4527,0.0272,0.9913,0.0570,
0.0791,0.9028,0.4706,0.4020,0.7592,0.4105,0.7107,0.5505,0.7223,0.3233,0.3037,0.9924,0.5545,
0.7944,0.0854,0.5545,0.4455,0.4636,0.2613,0.2467,0.9586,0.4275,0.8175,0.5793,0.1189,0.7109,
0.2115,0.8156,0.8468,0.9429,0.8382,0.1463,0.4212,0.6948,0.4816,0.3454,0.2114,0.3493,0.1389,
0.3448,0.0413,0.2422,0.6363,0.2340,0.8404,0.0065,0.7319,0.8781,0.2751,0.5197,0.4105,0.7121,
0.0832,0.1503,0.1148,0.3008,0.0121,0.0029,0.4423,0.6239,0.0651,0.3838,0.0165,0.2770,0.0475,
0.2074,0.0004,0.7962,0.4750,0.4839,0.9152,0.1681,0.0822,0.0518

Overall p-value after applying KStest on 269 p-values = 0.018449

Example 3. In this example as a source we used a highly biased source of randomness where 0 has frequency of $\frac{1}{1000}$ and 1 has frequency of $\frac{999}{1000}$. We applied several consecutive e-transformation with a random quasigroup of order

256, monitoring the results from Diehard battery. After the fifth e-transformation we obtained the following results:

```
** TEST SUMMARY - HIGHLY BIASED SOURCE & FIVE e-Transformations **
   All p-values:
0.9854,0.8330,0.4064,0.9570,0.6597,0.5447,0.5796,0.5885,0.3482,0.1359,0.1788,0.1194,0.8588,
0.3455,0.6627,0.3610,0.5622,0.9905,0.8430,0.1259,0.0799,0.9061,0.8378,0.4313,0.7249,0.4505,
0.9192,0.1007,0.2785,0.9099,0.0422,0.7891,0.2681,0.4452,0.9389,0.5081,0.7621,0.0914,0.0066,
0.6915,0.8662,0.7176,0.5658,0.7957,0.0590,0.4287,0.5772,0.4809,0.9891,0.1439,0.0000,0.6089,
0.2351,0.2533,0.0061,0.0171,0.6894,0.5279,0.9075,0.7313,0.6401,0.8004,0.1155,0.4374,0.8159,
0.9895,0.4989,0.5433,0.6915,0.9944,0.5661,0.7771,0.5461,0.8875,0.6586,0.0340,0.4701,0.9087,
0.1412,0.4037,0.7326,0.1809,0.3157,0.0573,0.3875,0.4210,0.9403,0.9805,0.2278,0.7588,0.2840,
0.5109,0.4997,0.5554,0.1334,0.5332,0.3025,0.2139,0.4366,0.2514,0.5530,0.7288,0.7055,0.3316,
0.0870,0.0853,0.6714,0.7704,0.9582,0.8772,0.2448,0.6751,0.0658,0.1317,0.6096,0.8317,0.0234,
0.6689,0.3353,0.5257,0.9411,0.7219,0.5881,0.1103,0.5709,0.3836,0.4470,0.6104,0.3517,0.5841,
0.1097,0.0597,0.6784,0.4045,0.6929,0.5104,0.5828,0.8125,0.5481,0.0264,0.3244,0.6821,0.8731,
0.8773,0.7624,0.7748,0.7128,0.4698,0.1195,0.0842,0.3780,0.8346,0.4562,0.5745,0.9541,0.3341,
0.0480,0.0753,0.3713,0.9637,0.9479,0.2401,0.8256,0.8368,0.2636,0.8346,0.9236,0.1218,0.3859,
0.8203,0.6748,0.5384,0.6346,0.8667,0.0006,0.6346,0.3780,0.8693,0.1459,0.7995,0.0483,0.7434,
0.2872,0.2546,0.2167,0.4233,0.8091,0.0451,0.2333,0.3243,0.8374,0.0915,0.3251,0.3731,0.5076,
0.8991,0.0931,0.9258,0.2831,0.8281,0.8386,0.0906,0.0979,0.5441,0.7129,0.8298,0.8427,0.8732,
0.7236,0.9397,0.5545,0.9397,0.9544,0.8312,0.2325,0.8424,0.2325,0.0176,0.8621,0.0401,0.7033,
0.2288,0.2786,0.6751,0.3424,0.5295,0.9344,0.7879,0.9744,0.0259,0.0487,0.1014,0.8589,0.8655,
0.1008,0.8204,0.5564,0.7432,0.8604,0.2008,0.2081,0.4452,0.2352,0.5092,0.4250,0.6055,0.5262,
0.1459,0.0838,0.2735,0.9764,0.6419,0.7941,0.2412,0.6055,0.9725,0.1075,0.2903,0.5552,0.1643,
0.0813,0.8206,0.0742,0.5889,0.3077,0.4771,0.7677,0.8252,0.3248
```

Overall p-value after applying KStest on 269 p-values = 0.373599

We now discuss the choice of the quasigroup, and the parameters s and k. If $E - algorithm$ and $E' - algorithm$ are used for simulating unbiased physical sources of randomness, then the quasigroup can be chosen to be arbitrary (we recommend $4 \leq s \leq 256$) while k depends on s and how biased is the source of randomness. The number k should be chosen by the rule 'for smaller s larger k' and its choice depends on the source. For example, if a source is highly biased (it produces 0 with probability 1/1000 and 1 with probability 999/1000), we suggest the following rule (derived from our numerous numerical experiments): '$ks \geq 512$ and $k > 8$'. In fact, the number s is in a way predefined by the source. Let the alphabet of the source consists of all 8-bits letters. Then we have the following choices of A: $A = \{0,1,2,3\}$, $A = \{0,1,2,\ldots,7\}$, $A = \{0,1,\ldots,15\}$, $A = \{0,1,\ldots,31\}$, $A = \{0,1,\ldots,63\}$, $A = \{0,1,2,\ldots,127\}$, $A = \{0,1,2,\ldots,255\}$. Thus, the output string of the source is considered as string of bits and then the bits are grouped in two, three, and so on. We can consider in this case alphabets with two–byte letters, three–byte letters etc., but quasigroups of orders 65536 or higher need a lot of storage memory and generally the computations are slower, and we do not recommend to be used.

If $E - algorithm$ and $E' - algorithm$ are used for improving some of the properties of PRNGs, then the quasigroup should be exponential. Our theoretical results ([8], [12], [13]) and numerical experiments indicate that the class of finite quasigroups can be separated into two disjoint subclasses: the class of linear quasigroups and the class of exponential quasigroups. There are several characteristics that separate these two classes and for our purposes this one is important. Given a finite set $Q = \{q_0, q_1, \ldots, q_{s-1}\}$, let $(Q, *)$ be a quasigroup and let $\alpha = q_0 q_1 \ldots\ q_{p-1} q_0 q_1 \ldots\ q_{p-1} q_0 q_1 \ldots\ q_{p-1} \ldots$ be an enough long string of period p. Let

$$\alpha_k = \underbrace{e_{l,*} \ldots e_{l,*}}_{k-\text{times}}(\alpha).$$

If the period of the string α_k is a linear function of k, then the quasigroup $(Q, *)$ is said to be linear. On the other hand, if the period of the string α_k is an exponential function $2^{c\,k}$ (where c is some constant), then the quasigroup $(Q, *)$ is said to be exponential. The number c is called the period growth of the exponential quasigroup $(Q, *)$.

The numerical experiments presented in [14] show that the percentage of linear quasigroups decreases when the order of the quasigroup increases. Furthermore, the percentage of 'bad' quasigroups, i.e. linear quasigroups and exponential quasigroup with period growth $c < 2$, is decreasing exponentially by the order of the quasigroups. For quasigroups of order 4, 5, 6, 7, 8, 9 and 10 the results are summarized in Table 3. We stress that the above results are not quite precise (except for the quasigroups of order 4, where complete classification is obtained in [15]), since the conclusion is made when only 7 e-transformation were applied. Namely, it can happen that some of quasigroups, after more than 7 applications, will have period growth $c \geq 2$.

Table 1. Percentage of 'bad' quasigroups of order 4 – 10

Order of the quasigroup	4	5	6	7	8	9	10
Percentage of 'bad' quasigroups	34.7	4.1	1.6	0.6	0.38	0.25	0.15

We made the following experiment over 10^6 randomly chosen quasigroups of order 16. We counted the period growth after 5 applications of $e_{l,*}$- transformations of each of the quasigroups on the following periodical strings with period 16: $0, 1, 2, \ldots, 14, 15, 0, 1, 2, \ldots, 14, 15, \ldots, 0, 1, 2, \ldots, 14, 15, \ldots$. The value of the leader l did not affect the results. The obtained distribution of the period growth is presented on the Table 2. It can be seen from Table 2 that 907 quasigroups have period growth $c < 2$ after 5 applications of the e-transformation. We counted the period growth after 6 applications of each of those quasigroups and we obtained that only 15 of them have period growth $c < 2$. After 7 applications, only one quasigroup has period growth $c < 2$, but after 10 applications of e-transformations, this quasigroup has period growth 2. This experiment shows that it is not easy to find randomly a linear quasigroup of order 16.

Table 2. Period growth of 10^6 randomly chosen quasigroups of order 16 after 5 applications of e-transformations

Value of c	Number of quasigroups with period growth $2^{c\,k}$	Value of c	Number of quasigroups with period growth $2^{c\,k}$
$0.00 \le c < 0.25$	4	$2.00 \le c < 2.25$	79834
$0.25 \le c < 0.50$	23	$2.25 \le c < 2.50$	128836
$0.50 \le c < 0.75$	194	$2.50 \le c < 2.75$	174974
$0.75 \le c < 1.00$	686	$2.75 \le c < 3.00$	199040
$1.00 \le c < 1.25$	2517	$3.00 \le c < 3.25$	175848
$1.25 \le c < 1.50$	7918	$3.25 \le c < 3.50$	119279
$1.50 \le c < 1.75$	18530	$3.50 \le c < 3.75$	45103
$1.75 \le c < 2.00$	42687	$3.75 \le c \le 4.00$	4527

4 Conclusion

We have suggested algorithms based on quasigroup string transformations for simulating unbiased coin flips using a biased source and for improving the properties of PRNGs. The performances of the algorithms are obtained from three theorems. The first theorem shows that the employed quasigroup string transformations are in fact permutations, the second theorem guarantees that the algorithms generate uniform output strings, while the third theorem proves that the period of the output pseudo random string can be arbitrary large number. We note that one have to choose an exponential quasigroup for obtaining better performances of the algorithms.

The proposed algorithms are very simple, of linear complexity and there are mathematical proofs of their properties. If quasigroups of order ≤ 16 are used the algorithms can be implemented in less than 1Kb working memory. Hence, they can be used in embedded systems as well.

The simplicity of the algorithms allows effective hardware realization. The initial results about parallel implementation of our algorithms are highly parallel and pipelined solution with delay of $O(n)$, where n is the number of e-transformations [16].

The use of the algorithms for cryptographic purposes (like designs of hash functions, synchronous, self-synchronizing and totaly asynchronous stream ciphers) is considered in several papers ([9], [17], [18], [19]), where it is emphasized that the employed quasigroups and the leaders of the transformations should be kept secret and the number n of applied e-transformations should be enough large. Note that the number of quasigroups of relatively small orders is huge one (there are 576 quasigroups of order 4, about 10^{20} of order 8, about 10^{47} of order 11 (see [20]), and much more than 10^{120} of order 16 and much more than 10^{58000} of order 256). On the other hand, by using the P. Hall's algorithm [21] for choosing a system of different representatives of a family of sets, a suitable algorithm for generating a random quasigroup of order s can be designed with complexity $O(s^3)$.

References

1. Gude, M.: Concept for a high-performance random number generator based on physical random phenomena. Frequenz, Vol. 39 (1985) 187–190
2. Agnew, G. B.: Random sources for cryptographic systems. In: D. Chaum and W. L. Price (eds.): Advances in Cryptology–Eurocrypt 87. Lecture Notes in Computer Science, Vol. 304. Springer–Verlag, Berlin Heidelberg New York (1987) 77–81
3. von Neumann, J.: Various techniques used in connection with random digits. In: Collected Works, Vol. 5. Pergamon Press (1963) 768–770
4. Elias, P.: The efficient construction of an unbiased random sequence. Ann. Math. Statist., Vol. 43, No. 3 (1972) 865–870
5. Stout, Q. F. and Warren, B. L.: Tree algorithms for unbiased coin tossing with a biased coin. Ann. Probab., Vol. 12, No. 1 (1984) 212–222
6. Juels, A., Jakobsson, M., Shriver, E. and Hillyer, B. K.: How to Turn Loaded Dice into Fair Coins. IEEE Transactions on Information Theory, Vol. 46, No. 3 (2000) 911–921
7. J. Dénes and A.D. Keedwell, Latin Squares and their Applications, English Univer. Press Ltd. (1974)
8. Markovski, S., Gligoroski, D., Bakeva, V.: Quasigroup String Processing: Part 1. Contributions, Sec. Math. Tech. Sci., MANU **XX**, 1-2 (1999) 13–28
9. Markovski, S.: Quasigroup string processing and applications in cryptography. Proc. 1-st Inter. Conf. Mathematics and Informatics for industry MII, Thessaloniki (2003) 278–290
10. http://stat.fsu.edu/~geo/diehard.html
11. http://www.nist.gov/
12. Markovski, S., Kusakatov, V.: Quasigroup String Processing: Part 2. Contributions, Sec. Math. Tech. Sci., MANU, **XXI**, 1-2 (2000) 15–32
13. Markovski, S., Kusakatov, V.: Quasigroup string processing: Part 3. Contributions, Sec. Math. Tech. Sci., MANU, **XXIII-XXIV**, 1-2 (2002-2003) 7–27
14. Dimitrova, V., Markovski, J.: On quasigroup pseudo random sequence generator. In: Manolopoulos, Y. and Spirakis, P. (eds.): Proc. of the 1-st Balkan Conference in Informatics, Thessaloniki (2004) 393–401
15. Markovski, S., Gligoroski, D., Markovski, J.: Classification of quasigroups by using random wolk on torus. Tech. Report, RISC-LINZ (2004) http://www.risc.uni-linz.ac.at/about/conferences/IJCAR-WS7
16. Gusev, M., Markovski,S., Gligoroski, G., Kocarev, Lj.: Processor Array Realization of Quasigroup String Transformations. Tech. Report, 01-2005, II-PMF Skopje (2005)
17. Markovski, S., Gligoroski, D., Bakeva, V.: Quasigroup and Hash Functions, Disc. Math. and Appl, Sl.Shtrakov and K. Denecke ed., Proceedings of the 6th ICDMA, Bansko 2001, pp . 43-50
18. Gligoroski, D., Markovski, S., Bakeva, V.: On Infinite Class of Strongly Collision Resistant Hash Functions "EDON-F" with Variable Length of Output. Proc. 1-st Inter. Conf. Mathematics and Informatics for industry MII, Thessaloniki (2003) 302–308
19. Gligoroski, G., Markovski,S., Kocarev, Lj.: On a family of stream ciphers "EDON". Tech. Report, 10-2004, II-PMF Skopje (2004)
20. McKay, B.D., Meynert, A. and Myrvold, W.: Small Latin Squares, Quaisgroups and Loops. http://csr.uvic.ca/~wendym/ls.ps
21. Hall, M.: Combinatorial theory. Blaisdell Publishing Company, Massachusetts (1967)

5 Appendix 1: Proof of Theorem 2

In order to simplify the technicalities in the proof we take that the alphabet A is $\{\mathbf{0},\dots,\mathbf{s-1}\}$, where $0,1,\dots,s-1$ $(s>1)$ are integers, and $*$ is a quasigroup operation on A. We define a sequence od random variables $\{Y_n|\ n\geq 1\}$ as follows. Let us have a probability distribution $(q_0,q_1,\dots,q_{s-1})$ of the letters $\mathbf{0},\mathbf{1},\dots,\mathbf{s-1}$, such that $q_i>0$ for each $i=0,1,\dots,s-1$ and $\sum\limits_{i=0}^{s-1} q_i=1$. Consider an e-transformation E and let $\gamma=E(\beta)$ where $\beta=b_1\dots b_k$, $\gamma=c_1\dots c_k\in A^+$ $(b_i,c_i\in A)$. We assume that the string β is arbitrarily chosen. Then by $\{Y_m=i\}$ we denote the random event that the m-th letter in the string γ is exactly $\mathbf{i}$. The definition of the e-transformation given by(3) implies

$$P(Y_m=j|\ Y_{m-1}=j_{m-1},\dots,Y_1=j_1)=P(Y_m=j|\ Y_{m-1}=j_{m-1})$$

since the appearance of the m-th member in γ depends only of the $(m-1)$-th member in γ, and not of the $(m-2)$-th,..., 1-st ones. So, the sequence $\{Y_m|\ m\geq 1\}$ is a Markov chain, and we refer to it as a quasigroup Markov chain (qMc). Let p_{ij} denote the probability that in the string γ the letter $\mathbf{j}$ appears immediately after the given letter $\mathbf{i}$, i.e.

$$p_{ij}=P(Y_m=j|\ Y_{m-1}=i),\qquad i,j=0,1,\dots,s-1.$$

The definition of qMc implies that p_{ij} does not depend of m, so we have that qMc is a homogeneous Markov chain. The probabilities p_{ij} can be determined as follows. Let $\mathbf{i},\mathbf{j},\mathbf{t}\in A$ and let $\mathbf{i}*\mathbf{t}=\mathbf{j}$ be a true equality in the quasigroup $(A,*)$. Then

$$P(Y_m=j|\ Y_{m-1}=i)=q_t,$$

since the equation $\mathbf{i}*x=\mathbf{j}$ has a unique solution for the unknown x. So, $p_{ij}>0$ for each $i,j=0,\dots,s-1$, i.e. the transition matrix $\Pi=(p_{ij})$ of qMc is regular. Clearly, as in any Markov chain, $\sum\limits_{j=0}^{s-1} p_{ij}=1$. But for the qMc we also have

$$\sum_{i=0}^{s-1} p_{ij}=\sum_{\mathbf{t}\in A} q_t=1$$

i.e. the transition matrix Π of a qMc is doubly stochastic.

As we have shown above, the transition matrix Π is regular and doubly stochastic. The regularity of Π implies that there is a unique fixed probability vector $p=(p_0,\dots,p_{s-1})$ such that $p\Pi=p$, and all components of p are positive. Also, since Π is a doubly stochastic matrix too, one can check that $\left(\frac{1}{s},\frac{1}{s},\dots,\frac{1}{s}\right)$ is a solution of $p\Pi=p$. So, $p_i=\frac{1}{s}$ $(i=0,\dots,s-1)$. In such a way we have the following Lemma:

Lemma 1. *Let $\beta = b_1 b_2 \dots b_k \in A^+$ and $\gamma = E^{(1)}(\beta)$. Then the probability of the appearance of a letter $\mathbf{i}$ at the m-th place of the string $\gamma = c_1 \dots c_k$ is approximately $\frac{1}{s}$, for each $\mathbf{i} \in A$ and each $m = 1, 2, \dots, k$.*

Lemma 1 tells us that the distribution of the letters in the string $\gamma = E(\beta)$ obtained from a sufficiently large string β by a quasigroup string permutation is uniform. We proceed the discussion by considering the distributions of the substrings $c_{i+1} \dots c_{i+l}$ of the string $\gamma = E^n(\beta)$ ($\beta = b_1 b_2 \dots b_k \in A^+$), where $l \geq 1$ is fixed and $i \in \{0, 1, \dots, k - l\}$. As usual, we say that $c_{i+1} \dots c_{i+l}$ is a substring of γ of length l. Define a sequence $\{Z_m^{(n)} \mid m \geq 1\}$ of random variables by

$$Z_m^{(n)} = t \Longleftrightarrow \begin{cases} Y_m^{(n)} = i_m^{(n)},\ Y_{m+1}^{(n)} = i_{m+1}^{(n)},\ \dots, Y_{m+l-1}^{(n)} = i_{m+l-1}^{(n)}, \\ t = i_m^{(n)} s^{l-1} + i_{m+1}^{(n)} s^{l-2} + \dots + i_{m+l-2}^{(n)} s + i_{m+l-1}^{(n)} \end{cases}$$

where here and further on the superscripts (n) denote the fact that we are considering substrings of a string $\gamma = \mathbf{i}_1^{(n)} \mathbf{i}_2^{(n)} \dots \mathbf{i}_k^{(n)}$ obtained from a string β by transformations of kind e^n. Thus, $Y_m^{(n)}$ is just the random variable Y_m defined as before. The mapping

$$(\mathbf{i}_m^{(n)}, \mathbf{i}_{m+1}^{(n)}, \dots, \mathbf{i}_{m+l-1}^{(n)}) \ \mapsto\ i_m^{(n)} s^{l-1} + i_{m+1}^{(n)} s^{l-2} + \dots + i_{m+l-2}^{(n)} s + i_{m+l-1}^{(n)}$$

is a bijection from A^l onto $\{0, 1, \dots, s^l - 1\}$, so the sequence $\{Z_m^{(n)} \mid m \geq 1\}$ is well defined. The sequence $\{Z_m^{(n)} \mid m \geq 1\}$ is also a Markov chain (n-qMc), since the appearance of a substring $\mathbf{i}_m^{(n)} \mathbf{i}_{m+1}^{(n)} \dots \mathbf{i}_{m+l-1}^{(n)}$ of l consecutive symbols in γ depends only of the preceding substring $\mathbf{i}_{m-1}^{(n)} \mathbf{i}_m^{(n)} \mathbf{i}_{m+1}^{(n)} \dots \mathbf{i}_{m+l-2}^{(n)}$. Denote by t and t' the following numbers:

$$t = i_m^{(n)} s^{l-1} + i_{m+1}^{(n)} s^{l-2} + \dots + i_{m+l-2}^{(n)} s + i_{m+l-1}^{(n)},$$

$$t' = i_{m-1}^{(n)} s^{l-1} + i'^{(n)}_m s^{l-2} + \dots + i'^{(n)}_{m+l-3} s + i'^{(n)}_{m+l-2}.$$

Let $p_{t't}$ be the probability that in some string $\gamma = E^{(n)}(\beta)$, the substring $\mathbf{i}_m^{(n)} \dots \mathbf{i}_{m+l-2}^{(n)} \mathbf{i}_{m+l-1}^{(n)}$ of γ (from the m-th to the $m + l - 1$-th position) appears (with overlapping) after a given substring $\mathbf{i}_{m-1}^{(n)} \mathbf{i}'^{(n)}_m \dots \dots \mathbf{i}'^{(n)}_{m+l-3} \mathbf{i}'^{(n)}_{m+l-2}$ of γ (from the $m - 1$-th to the $m + l - 2$-th position). Clearly, $p_{t't} = 0$ if $\mathbf{i}_j^{(n)} \neq \mathbf{i}'^{(n)}_j$ for some $j \in \{m, m-1, \dots, m + l - 2\}$. In the opposite case (when $l - 1$ letters are overlapped) we have:

$$
\begin{aligned}
p_{t't} &= P(Z_m^{(n)} = t \mid Z_{m-1}^{(n)} = t') \\
&= P(Y_m^{(n)} = i_m^{(n)}, \dots, Y_{m+l-1}^{(n)} = i_{m+l-1}^{(n)} \mid Y_{m-1}^{(n)} = i_{m-1}^{(n)}, Y_m^{(n)} = i_m^{(n)}, \dots \\
&\qquad\qquad \dots, Y_{m+l-2}^{(n)} = i_{m+l-2}^{(n)}) \\
&= P(\cap_{j=0}^{l-1} (Y_{m+j}^{(n)} = i_{m+j}^{(n)}) \mid \cap_{j=0}^{l-1} (Y_{m+j-1}^{(n)} = i_{m+j-1}^{(n)})) \\
&= \frac{P(\cap_{j=0}^{l} (Y_{m+j-1}^{(n)} = i_{m+j-1}^{(n)}))}{P(\cap_{j=0}^{l-1} (Y_{m+j-1}^{(n)} = i_{m+j-1}^{(n)}))} \\
&= \frac{P(\cap_{j=0}^{l-1} (Y_{m+j}^{(n)} = i_{m+j}^{(n)})) \mid Y_{m-1}^{(n)} = i_{m-1}^{(n)})}{P(\cap_{j=0}^{l-2} (Y_{m+j}^{(n)} = i_{m+j}^{(n)})) \mid Y_{m-1}^{(n)} = i_{m-1}^{(n)})}
\end{aligned}
\tag{5}
$$

By using an induction of the numbers n of quasigroup transformations we will prove the Theorem 2, i.e we will prove the following version of it:

Let $1 \le l \le n,\ \beta = b_1 b_2 \dots b_k \in A^+$ and $\gamma = E^{(n)}(\beta)$. Then the distribution of substrings of γ of length l is uniform.

Recall the notation $A = \{\mathbf{0}, \dots, \mathbf{s-1}\}$. For $n = 1$ we have the Lemma 1, and let $n = r + 1,\ r \ge 1$. By the inductive hypothesis, the distribution of the substrings of length l for $l \le r$ in $\gamma' = E^r(\beta)$ is uniform. At first, we assume $l \le r$ and we are considering substrings of length l of $\gamma = E^{r+1}(\beta) = \mathbf{i}_1^{(r+1)} \dots \mathbf{i}_k^{(r+1)}$. We take that $*_1, \dots, *_{r+1}$ are quasigroup operations on A and recall that $E^{(r+1)} = E_{r+1} \circ E^{(r)} = E_{r+1} \circ E_r \circ E^{(r-1)} = \dots$. Since $(A, *_{r+1})$ is a quasigroup, the equation $\mathbf{i}_{j-1}^{(r+1)} *_{r+1} \mathbf{x} = \mathbf{i}_j^{(r+1)}$ has a unique solution on $\mathbf{x}$, for each $j,\ 2 \le j \le k$, and we denote it by $\mathbf{x} = \mathbf{i}_j^{(r)}$. Denote by $\mathbf{i}_1^{(r)}$ the solution of the equation $a_{r+1} *_{r+1} \mathbf{x} = \mathbf{i}_1^{(r+1)}$, where $a_{r+1} \in A$ is the fixed element in the definition of E_{r+1}. In such a way, instead of working with substrings $\mathbf{i}_m^{(r+1)} \mathbf{i}_{m+1}^{(r+1)} \dots \mathbf{i}_{m+d}^{(r+1)}$ of γ, we can consider substrings $\mathbf{i}_m^{(r)} \mathbf{i}_{m+1}^{(r)} \dots \mathbf{i}_{m+d}^{(r)}$ of $\gamma' = E^{(r)}(\beta)$, for any $d,\ 0 \le d \le k - m$. The uniqueness of the solutions in the quasigroup equations implies that we have

$$
P(\cap_{j=0}^{d} (Y_{m+j}^{(r+1)} = i_{m+j}^{(r+1)})) \mid Y_{m-1}^{(r+1)} = i_{m-1}^{(r+1)}) = P(\cap_{j=0}^{d} (Y_{m+j}^{(r)} = i_{m+j}^{(r)})) \tag{6}
$$

as well. Here, $i_0^{(r+1)} = a_{r+1}$. Then, by (5) and (6) (for $d = l - 1$, $d = l - 2$ and $n = r + 1$) we have

$$
p_{t't} = \frac{P(\cap_{j=0}^{l-1} (Y_{m+j}^{(r)} = i_{m+j}^{(r)}))}{P(\cap_{j=0}^{l-2} (Y_{m+j}^{(r)} = i_{m+j}^{(r)}))} \tag{7}
$$

where $l \leq r$. By the inductive hypothesis we have $P(\cap_{j=0}^{l-1}(Y_{m+j}^{(r)} = i_{m+j}^{(r)})) = \frac{1}{s^l}$, $P(\cap_{j=0}^{l-2}(Y_{m+j}^{(r)} = i_{m+j}^{(r)})) = \frac{1}{s^{l-1}}$, i.e. $p_{t't} = \frac{1}{s}$. Thus, for the probabilities $p_{t't}$ we have

$$p_{t't} = \begin{cases} 0 & \text{if } \mathbf{i'}_j^{(r+1)} \neq \mathbf{i}_j^{(r+1)} \text{ for some } j = m, \dots, m+l-2 \\ \frac{1}{s} & \text{if } \mathbf{i'}_j^{(r+1)} = \mathbf{i}_j^{(r+1)} \text{ for each } j = m, \dots, m+l-2. \end{cases}$$

This means that in each column of the $s^l \times s^l$-matrix of transitions Π of n-qMc there will be exactly s members equal to $\frac{1}{s}$ (those for which $\mathbf{i'}_j^{(r+1)} = \mathbf{i}_j^{(r+1)}$, $j = m, \dots, m+l-2$), the other members will be equal to 0 and then the sum of the members of each column of Π is equal to 1. Hence, the transition matrix Π is doubly stochastic. It is a regular matrix too, since each element of the matrix Π^l is positive. This implies that the system $p\Pi = p$ has a unique fixed probability vector $p = \left(\frac{1}{s^l}, \frac{1}{s^l}, \dots, \frac{1}{s^l}\right)$ as a solution. In other words, the distribution of substrings of γ of length $l \leq r$ is uniform. Assume now that $l = r+1$, and let the numbers t, t' and the probabilities $p_{t't}$ be defined as before. Then for $p_{t't}$ we have that (7) holds too, i.e.

$$p_{t't} = \frac{P(\cap_{j=0}^{r}(Y_{m+j}^{(r)} = i_{m+j}^{(r)}))}{P(\cap_{j=0}^{r-1}(Y_{m+j}^{(r)} = i_{m+j}^{(r)}))} = \frac{P(\cap_{j=0}^{r-1}(Y_{m+j+1}^{(r)} = i_{m+j+1}^{(r)}) \mid Y_m^{(r)} = i_m^{(r)})}{P(\cap_{j=0}^{r-2}(Y_{m+j+1}^{(r)} = i_{m+j+1}^{(r)}) \mid Y_m^{(r)} = i_m^{(r)})} \tag{8}$$

In the same way as it was done before, by using the fact that the equations $\mathbf{i}_{j-1}^{(u)} *_u \mathbf{x} = \mathbf{i}_j^{(u)}$ have unique solutions $\mathbf{x} = \mathbf{i}_j^{(u-1)}$ in the quasigroup $(A, *_u)$, where $u = r, r-1, \dots, 2, 1$, we could consider substrings of $\gamma' = E^{(r)}(\beta)$, $\gamma'' = E^{(r-1)}(\beta)$, $\dots$, $\gamma^{(r)} = E^{(1)}(\beta)$, $\gamma^{(r+1)} = E^{(0)}(\beta) = \beta$. Then, for the probabilities $p_{t't}$, by repeatedly using the equations (6) and (8), we will reduce the superscripts (r) to $(r-1)$, to $(r-2)$, $\dots$, to (1), i.e. we will have

$$\begin{aligned} p_{t't} &= \frac{P(Y_{m+r-1}^{(1)} = i_{m+r-1}^{(1)},\ Y_{m+r}^{(1)} = i_{m+r}^{(1)})}{P(Y_{m+r-1}^{(1)} = i_{m+r-1}^{(1)})} \\ &= P(Y_{m+r}^{(1)} = i_{m+r}^{(1)} \mid Y_{m+r-1}^{(1)} = i_{m+r-1}^{(1)}) \\ &= P(Y_{m+r}^{(0)} = i_{m+r}^{(0)}) \end{aligned}$$

where $\mathbf{i}_{m+r}^{(0)} \in \beta$. Since $P(Y_{m+r}^{(0)} = i_{m+r}^{(0)}) = q_{i_{m+r}}^{(0)}$ we have

$$p_{t't} = \begin{cases} 0 & \text{if } \mathbf{i'}_j^{(r+1)} \neq \mathbf{i}_j^{(r+1)} \text{ for some } j = m, \dots, m+r-1 \\ q_{i_{m+r}}^{(0)} & \text{if } \mathbf{i'}_j^{(r+1)} = \mathbf{i}_j^{(r+1)} \text{ for each } j = m, \dots, m+r-1 \end{cases}$$

which implies

$$\begin{aligned}\sum_{t'=0}^{s^{r+1}-1} p_{t't} &= \sum_{i_{m-1}^{(r+1)}=0}^{s-1} \sum_{i'^{(r+1)}_{m}=0}^{s-1} \cdots \sum_{i'^{(r+1)}_{m+r-2}=0}^{s-1} p_{t't} = \sum_{i_{m-1}^{(r+1)}=0}^{s-1} q_{i_{m+r}}^{(0)} \\ &= \sum_{i_{m}^{(r)}=0}^{s-1} q_{i_{m+r}}^{(0)} = \sum_{i_{m+1}^{(r-1)}=0}^{s-1} q_{i_{m+r}}^{(0)} = \cdots = \sum_{i_{m+r}^{(0)}=0}^{s-1} q_{i_{m+r}}^{(0)} = 1\end{aligned} \tag{9}$$

We should note that the equations

$$\sum_{i_{m-1}^{(r+1)}=0}^{s-1} q_{i_{m+r}}^{(0)} = \sum_{i_{m}^{(r)}=0}^{s-1} q_{i_{m+r}}^{(0)} = \dots$$

hold true since the equations $\mathbf{i}_{j-1}^{(u)} *_u \mathbf{x} = \mathbf{i}_{j}^{(u)}$ have unique solutions in the quasigroup $(A, *_u)$, for each $u = r+1, r, \dots, 2, 1$.
Hence, the transition matrix Π is doubly stochastic, it is regular (Π^{r+1} has positive entries) which means that the system $p\Pi = p$ has a unique fixed probability vector $p = \left(\frac{1}{s^{r+1}}, \frac{1}{s^{r+1}}, \dots, \frac{1}{s^{r+1}}\right)$ as a solution.

Remark 1. Generally, the distribution of the substrings of lengths l for $l > n$ in a string $\gamma = E^{(n)}(\beta)$ is not uniform. Namely, for $l = n+1$, in the same manner as in the last part of the preceding proof, one can show that $p_{t't} = P(Y_{m+n+1}^{(0)} = i_{m+n+1}^{(0)} \mid Y_{m+n}^{(0)} = i_{m+n}^{(0)})$ and then (as in (9)) we have

$$\sum_{t'=0}^{s^{n+1}-1} p_{t't} = \sum_{i_{m+n}^{(0)}=0}^{s-1} P(Y_{m+n+1}^{(0)} = i_{m+n+1}^{(0)} \mid Y_{m+n}^{(0)} = i_{m+n}^{(0)}).$$

Of course, the last sum must not be equal to 1, i.e. the transition matrix Π must not be doubly stochastic. The same consideration could be made for $l = n+2, n+3, \dots$ as well.

6 Appendix 2: Proof of Theorem 3

Consider a finite quasigroup $(A, *)$ of order s and take a fixed element $a \in A$ such that $a * a \neq a$. We will prove the Theorem 3 in the more extreme case and so we take a string $\alpha = a_1 \dots a_k$ of period 1 where $a_i = a$ for each $i \geq 1$. Then we apply the transformation $E = e_{a,*}$ on α several times. E^n means that E is applied n times and we denote $E^n(\alpha) = a_1^{(n)} \dots a_k^{(n)}$. The results are presented on Figure 6. We have that $a'_p = a$ for some $p > 1$ since $a * a \neq a$ and $a'_i \in A$ (so

$$\begin{array}{c|cccccc}
 & a & a & \dots & a & a & \dots \\
\hline
a & a_1^{'} & a_2^{'} & \dots & a_{p-1}^{'} & a_p^{'} & \dots \\
 & & & & & & \\
a & a_1^{''} & a_2^{''} & \dots & a_{p-1}^{''} & a_p^{''} & \dots \\
 & & & & & & \\
a & a_1^{'''} & a_2^{'''} & \dots & a_{p-1}^{'''} & a_p^{'''} & \dots \\
 & & & & & & \\
a & a_1^{(4)} & a_2^{(4)} & \dots & a_{p-1}^{(4)} & a_p^{(4)} & \dots \\
\vdots & \vdots & \vdots & & \vdots & \vdots &
\end{array}$$

we have that p is at least s), and let p be the smallest integer with this property. It follows that the string $E(\alpha)$ is periodical with period p. For similar reasons we have that each of the strings $E_a^n(\alpha)$ is periodical. We will show that it is not possible all of the strings $E_a^n(\alpha)$ to be of same period p . If we suppose that it is true, we will have $a_p^{(n)} = a$ for each $n \geq 1$. Then we will also have that there are $b_i \in A$ such that the following equalities hold:

$$\begin{array}{ll}
a_{p-1}^{(n)} = b_{p-1} & \text{for } n \geq 2 \\
a_{p-2}^{(n)} = b_{p-2} & \text{for } n \geq 3 \\
\vdots & \\
a_1^{(n)} = b_1 & \text{for } n \geq p
\end{array}$$

Then we have that $a * b_1 = b_1$, and that implies $a_1^{(n)} = b_1$ for each $n \geq 1$. We obtained $a * a = a * b_1 = b_1$, implying $a = b_1$, a contradiction with $a * a \neq a$. As a consequence we have that $a_1^{(p+1)} = a * a_1^{(p)} = a * b_1 \neq b_1,\ a_2^{(p+1)} = a_1^{(p+1)} * b_2 \neq b_2,\ \dots, a_{p-1}^{(p+1)} = a_{p-2}^{(p+1)} * b_{p-1} \neq b_{p-1},\ a_p^{(p+1)} = a_{p-1}^{(p+1)} * a \neq a$. We conclude that the period of the string $E_a^{p+1}(\alpha)$ is not p.

Next we show that if a string $\beta \in A^+$ has a period p and $\gamma = E(\beta)$ has a period q, then p is a factor of q. Recall that the transformation E by Theorem 1 is a permutation and so there is the inverse transformation E^{-1}. Now, if $\gamma = b_1 \dots b_q b_1 \dots b_q \dots b_1 \dots b_q$, then $\beta = E^{-1}(\gamma) = c_1 c_2 \dots c_q c_1 c_2 \dots c_q \dots c_1 c_2 \dots c_q$ is a periodical string with period $\leq q$. So, $p \leq q$ and this implies that p is a factor of q.

Combining the preceding results, we have proved the following version of Theorem 3:

Let α be a string with period p_0. Then the strings $\beta = E_a^n(\alpha)$ are periodical with periods p_n that are multiples of p_0. The periods p_n of β satisfy the inequality

$$p_{p_{n-1}} > p_{n-1}$$

for each $n \geq 1$.