Infinite words with well distributed occurrences

Ľ. Balková, M. Bucci, J. Hladký, A. De Luca, S. Puzynina

WORDS 2013 in Turku

September, 20

Program

1. PRNGs based on infinite words
2. Words with well distributed occurrences
3. Statistical tests

Linear congruential generator (LCG)

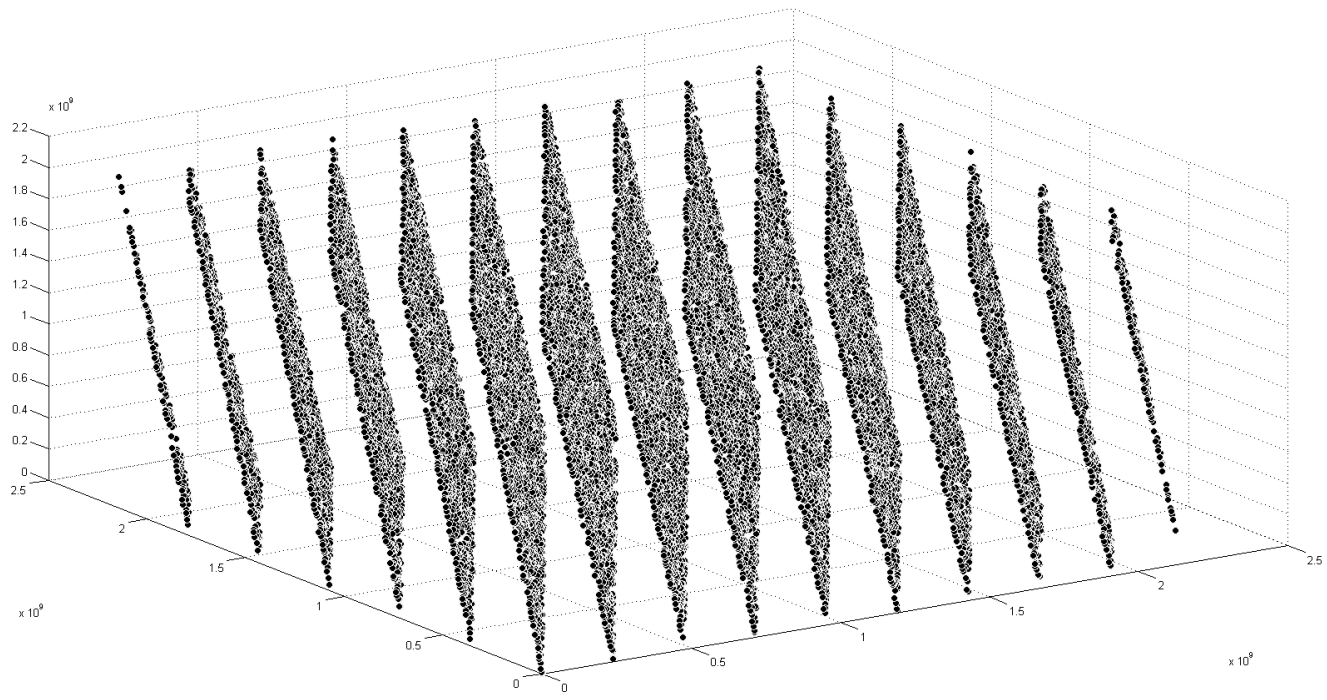
- defined by a recurrent relation

$$x_{n+1} = (ax_n + c) \mod m$$

$0 < m$	modulo
$0 \leq a < m$	multiplier
$0 \leq c < m$	shift
$0 \leq x_0 < m$	seed

- weaknesses of LCGs:
  - periodicity
  - lattice structure

Lattice structure of LCGs



RANDU: $a = 65539$, $m = 2^{31}$, $c = 0$

Program

Binary words

- we are given two LCGs with modulo $m \geq 2$ and output $\{0, 1, \ldots, m-1\}$

$$X = (x_n)_{n \geq 0} \text{ a } Y = (y_n)_{n \geq 0}$$

and an infinite word $\mathbf{u} = u_1 u_2 u_3 \ldots$ over $\{0, 1\}$

- then the PRNG $Z = (z_n)_{n \geq 0}$ based on $\mathbf{u}$ is obtained by the following algorithm:
  1. we read letters of $\mathbf{u}$
  2. if we read in $\mathbf{u}$ a zero for the i-th time, then we add to the sequence Z the i-th element of X
  3. if we read in $\mathbf{u}$ a one for the i-th time, then we add to the sequence Z the i-th element of Y

Properties of PRNGs based on infinite words

- PRNG keeps aperiodicity
- PRNG based on a subclass of "cut and project" sequences do not have the lattice structure

Guimond L.-S., Patera J., Patera J., *Statistical properties and implementation of aperiodic pseudorandom number generators*, Applied Numerical Mathematics **46(3-4)** (2003), 295–318

Program

Absence of the lattice structure

- PRNGs based on words with well distributed occurrences do not have the lattice structure
- an aperiodic infinite word $\mathbf{u}$ over $\{0,1\}$ has *well distributed occurrences (the WDO property)* if for every $m \in \mathbb{N}$ and for every factor w the following condition is satisfied: denote $i_1, i_2, \ldots$ occurrences of w in $\mathbf{u}$, then

$$\left\{ (|u_1 u_2 \ldots u_{i_j}|_0, |u_1 u_2 \ldots u_{i_j}|_1) \mod m \;\middle|\; j \in \mathbb{N} \right\} = \mathbb{Z}_m^2$$

- example: the Thue-Morse word does not have the WDO property

Absence of the lattice structure

- PRNGs based on words with well distributed occurrences do not have the lattice structure
- an aperiodic infinite word $\mathbf{u}$ over $\{0,1\}$ has *well distributed occurrences (the WDO property)* if for every $m \in \mathbb{N}$ and for every factor w the following condition is satisfied: denote $i_1, i_2, \ldots$ occurrences of w in $\mathbf{u}$, then

$$\left\{ (|u_1 u_2 \ldots u_{i_j}|_0, |u_1 u_2 \ldots u_{i_j}|_1) \mod m \;\middle|\; j \in \mathbb{N} \right\} = \mathbb{Z}_m^2$$

- example: the Thue-Morse word does not have the WDO property

Sturmian words

- Sturmian words have the WDO property
- they form a larger class than the subclass of “cut and project” sequences considered in the paper

Multiliteral alphabets

- if we combine more LCGs, then the obtained PRNG does not have the lattice structure if the corresponding infinite word $\mathbf{u}$ has the WDO property:
- an aperiodic word $\mathbf{u}$ over $\{0, 1, \ldots, d-1\}$ has the *WDO property* if for every $m \in \mathbb{N}$ and every factor w the following condition is satisfied: denote $i_1, i_2, \ldots$ occurrences of w in $\mathbf{u}$, then

$$\left\{ \left(|u_1 u_2 \ldots u_{i_j}|_0, \ldots, |u_1 u_2 \ldots u_{i_j}|_{d-1} \right) \mod m \;\middle|\; j \in \mathbb{N} \right\} = \mathbb{Z}_m^d$$

- the universal word has the WDO property

Multiliteral alphabets

- if we combine more LCGs, then the obtained PRNG does not have the lattice structure if the corresponding infinite word $\mathbf{u}$ has the WDO property:
- an aperiodic word $\mathbf{u}$ over $\{0, 1, \ldots, d-1\}$ has the *WDO property* if for every $m \in \mathbb{N}$ and every factor w the following condition is satisfied: denote $i_1, i_2, \ldots$ occurrences of w in $\mathbf{u}$, then

$$\left\{ \left(|u_1 u_2 \ldots u_{i_j}|_0, \ldots, |u_1 u_2 \ldots u_{i_j}|_{d-1} \right) \mod m \;\middle|\; j \in \mathbb{N} \right\} = \mathbb{Z}_m^d$$

- the universal word has the WDO property

Arnoux-Rauzy words

- an infinite word $\mathbf{u}$ is called *Arnoux-Rauzy* over $\{0, 1, \ldots, d-1\}$ if its language is closed under reversal and $\mathbf{u}$ has one LS factor of every length and this factor has exactly d left extensions
- AR words have the WDO property
- example: if we replace $d-1$ with 0 in an AR word, the WDO property is kept

Arnoux-Rauzy words

- an infinite word $\mathbf{u}$ is called *Arnoux-Rauzy* over $\{0, 1, \ldots, d-1\}$ if its language is closed under reversal and $\mathbf{u}$ has one LS factor of every length and this factor has exactly d left extensions
- AR words have the WDO property
- example: if we replace $d-1$ with 0 in an AR word, the WDO property is kept

Program

TestU01, PractRand

- excellent results in tests (in comparison to LCGs)
- runtime penalization is negligible, moreover a trade-off between memory footprint and speed of generation of the infinite words is possible

TestU01, PractRand

- excellent results in tests (in comparison to LCGs)
- runtime penalization is negligible, moreover a trade-off between memory footprint and speed of generation of the infinite words is possible
- multiliteral alphabet is better

TestU01, PractRand

- excellent results in tests (in comparison to LCGs)
- runtime penalization is negligible, moreover a trade-off between memory footprint and speed of generation of the infinite words is possible
- multiliteral alphabet is better
- combining LCGs of different quality does not seem to be useful

TestU01, PractRand

- excellent results in tests (in comparison to LCGs)
- runtime penalization is negligible, moreover a trade-off between memory footprint and speed of generation of the infinite words is possible
- multiliteral alphabet is better
- combining LCGs of different quality does not seem to be useful
- using LCGs with the same multiplier and with distinct seeds provides as good results as combination of LCGs with different multipliers (modulo m is always the same)

TestU01, PractRand

- excellent results in tests (in comparison to LCGs)
- runtime penalization is negligible, moreover a trade-off between memory footprint and speed of generation of the infinite words is possible
- multiliteral alphabet is better
- combining LCGs of different quality does not seem to be useful
- using LCGs with the same multiplier and with distinct seeds provides as good results as combination of LCGs with different multipliers (modulo m is always the same)

Questions

- Why PRNGs based on infinite words with the WDO property succeed in statistical tests?
- Is there any relation between other combinatorial properties of infinite words (complexity, palindromes, factor frequencies, return words etc.) and statistical properties of arising PRNGs?

Questions

- Why PRNGs based on infinite words with the WDO property succeed in statistical tests?
- Is there any relation between other combinatorial properties of infinite words (complexity, palindromes, factor frequencies, return words etc.) and statistical properties of arising PRNGs?

Generator	Time(10^{10})	BigCrush	PractRand
LCG($2^{47}-115, 71971110957370, 0$)	281	14	1TB
LCG($2^{63}-25, 2307085864, 0$)	277	2	>16TB
LCG($2^{59}, 13^{13}, 0$)	14.1	19	128MB
LCG($2^{63}, 5^{19}, 1$)	14.4	19	8GB
LCG($2^{63}, 9219741426499971445, 1$)	14.4	19	8GB
LCG($2^{64}, 2862933555777941757, 1$)	14.0	18	32GB
LCG($2^{64}, 3202034522624059733, 1$)	14.1	14	16GB
LCG($2^{64}, 3935559000370003845, 1$)	14.0	13	8GB

Generator	Time(10^{10})	BigCrush	PractRand	Combination
Fibonacci	25.7	(8×) 0 (0) (1×) 0 (1)	(9×) 2TB	Total 9 combinations {1,0}
Fibonacci2	17.3	(22×) 0 (0) (4×) 0 (1) (1×) 0 (2)	(25×) 1TB (2×) 0.5TB	Total 27 combinations {0,1,1} {0,2,1} {0,1,2} {0,2,2} {0,0,0} {0,0,2}
Tribonacci	25.7	(22×) 0 (0) (4×) 0 (1) (1×) 0 (2)	(16×) 2TB (7×) 4TB (2×) 8TB (2×) 1TB	Total 27 combinations {1,1,0} {2,1,0} {1,2,0} {2,2,0} {0,0,0} {0,0,0} {0,1,1} {0,2,1} {0,1,2} {0,2,2} {0,1,0} {0,2,0} {0,0,2} {0,0,1} {1,0,0} {2,0,0}

Thank you for attention