УДК 004.421.5:511.174
DOI: 10.31673/2409-7292.2024.020011

Горячий О. Я., Максимович В. М., Шабатура М. М.

ДОСЛІДЖЕННЯ МНОЖИНИ ПОЧАТКОВИХ ЗНАЧЕНЬ ГЕНЕРАТОРІВ ПСЕВДОВИПАДКОВИХ ЧИСЕЛ НА ОСНОВІ АРИФМЕТИКИ З РУХОМОЮ КОМОЮ

У даній статті досліджено статистичні характеристики та період повторення запропонованих раніше генераторів псевдовипадкових чисел (ГПВЧ), побудованих на основі наближених методів обчислення елементарних функцій в арифметиці з рухомою комою. Метою роботи була оцінка якості згенерованих послідовностей, визначення допустимих діапазонів початкових значень та вибору параметрів генератора із найкращими характеристиками. Досліджено використання тотожної функції, алгоритму наближеного обчислення оберненого значення та шести алгоритмів зворотного квадратного кореня у двох режимах генерації та для п’яти можливих комбінацій параметрів ГПВЧ залежно від вибору початкових значень. Для тестування запропонованих варіантів генераторів формувались послідовності псевдовипадкових чисел розміру 2 Гб для фіксованого набору їх початкових значень. З’ясовано, що період повторення залежить від вибору початкового значення генератора, параметру *bytesCount* та елементарної функції. Виконано статистичне тестування послідовностей на основі набору тестів NIST, графічного тесту та аналізу гістограм. Запропоновано діапазони вибору початкових значень залежно від параметрів генератора та вибрано три генератори із найкращими характеристиками. Зокрема, більш як 82% сформованих послідовностей генератора *rsqrt2dc_everyBit* з параметрами *bytesCount* = 1 та *offse*t = 0 (максимальний період – 222,12 Мб) проходять всі тести NIST. Встановлено, що для вибору початкових значень оптимальною є множина розмірності близько 2^{60} елементів, $x_0 \in [0{,}5;\ 10^{135})$. Виявлено певні статистичні закономірності та недоліки окремих генераторів. Додатково продемонстровано, що алгоритми апроксимації елементарних функцій в арифметиці з рухомою комою можуть використовуватись для побудови ГПВЧ із непоганими характеристиками.

Ключові слова: псевдовипадкова послідовність, генератори псевдовипадкових чисел, елементарна функція, апроксимаційні методи, арифметика з рухомою комою, статистичні тести NIST, період повторення.

Вступ

Генератори псевдовипадкових чисел (ГПВЧ) мають безліч як криптографічних, так і не криптографічних застосувань. Вони формують детерміновані послідовності, що повністю визначаються параметрами відповідного генератора та його початковими значеннями, а тому повинні забезпечувати хороші характеристики згенерованих послідовностей для будь-якого початкового значення [1–3].

Для прикладу, псевдовипадкові числа (ПВЧ) або псевдовипадкові бітові послідовності (ПВБП) використовуються в стеганографічних алгоритмах, під час роботи скремблерів та потокових шифрів, під час виконання комп’ютерних симуляцій, реалізації алгоритмів комп’ютерної графіки та машинного навчання [2–8]. Для швидкої генерації таких псевдовипадкових послідовностей розробляються спеціальні алгоритми та відповідні програмні чи апаратні пристрої, що називаються генераторами ПВЧ або ПВБП. Ключовими перевагами таких генераторів у порівнянні з генераторами випадкових послідовностей є їх простота реалізації, суттєво вища швидкодія та можливість відтворення сформованих ними послідовностей на приймаючій стороні або при повторній симуляції. Вони зазвичай мають скінченний період повторення та за своїми властивостями подібні до істинно випадкових, що є непередбачуваними, неперіодичними та які неможливо відтворити без попереднього запису [1].

Іншими прикладами використання ГПВЧ є формування криптографічних ключів, векторів ініціалізації, одноразових кодів (nonce) та значень солі (salt) для протоколів автентифікації, випадкових значень у схемах доповнення, в протоколах захищеного багатокористувацького обчислення (multi-party computation) та розділення секрету (secret sharing), в комп’ютерних іграх, в безпровідних системах передачі даних, в числових методах на основі методу Монте-Карло, а також під час моделювання різноманітних фізичних, біологічних та соціальних процесів [1, 2, 8].

Постановка проблеми

Класичні алгоритми генерації ПВЧ базуються переважно на цілочислових операціях, зокрема взаємовиключному АБО, бітовому зсуві, знаходженні остачі від цілочислового ділення тощо. На сучасних пристроях із математичним співпроцесором (FPU) базові операції з рухомою комою стандарту IEEE 754 [9] (+, –, ×, fma) також виконуються досить швидко. Проте, в літературі мало уваги приділено використанню швидких наближених алгоритмів обчислення елементарних функцій, таких як $1/x$, $\sqrt{x}$ та $1/\sqrt{x}$, в арифметиці з рухомою комою для проектування ГПВЧ із хорошими статистичними характеристиками.

З практичної точки зору під час проектування ГПВЧ мають важливе значення наступні вимоги: висока швидкість генерації, рівномірний розподіл згенерованих числових послідовностей, великий період повторення та непередбачуваність послідовностей ПВЧ [1, 7].

Аналіз наукових публікацій

Популярними типами алгоритмів для генерації ПВЧ є ГПВЧ на основі регістрів зсуву з лінійним зворотним зв'язком [1, 4, 8, 10], лінійного конгруентного методу [1, 8], вихору Мерсена [1, 3, 7, 8, 11], віднімання із запозиченням, множення із переносом [12] тощо. Ці генератори можуть мати величезний період повторення та хороші статистичні властивості, проте є непридатними для криптографічних застосувань [1, 7]. Серед криптографічно стійких генераторів можна виділити генератори Блум-Блум-Шуба [3, 13] та Фортуна [1], а також інші ГПВЧ на основі односторонніх функцій, блокових і потокових шифрів та криптографічних хеш-функцій [1]. Проте, недоліком таких алгоритмів є їх низька швидкодія.

На сьогодні набувають популярності також ГПВЧ на основі теорії хаосу, що мають достатній рівень продуктивності та криптографічної стійкості [6, 7, 14]. Перспективність досліджень таких генераторів базується на основі нелінійності та хаотичної поведінки логістичного рівняння [6], хаотичних або гіперхаотичних систем [5, 14]. Як можна побачити, під час реалізації таких ГПВЧ на практиці, як правило, використовуються числа у форматі з рухомою комою.

В наукових колах та в середовищі масової культури досі популярною є дискусія щодо нормальності та статистичної випадковості цифр десяткового/двійкового представлення певних популярних математичних констант, таких як π=3,1415926535… та e =2,7182818284… [12, 15, 16], а також квадратних коренів із простих чисел, таких як $\sqrt{2}$ =1,4142135623…, $\sqrt{3}$ =1,7320508075… та ін. [12, 16, 17]. Прийнято вважати, що такі послідовності проходять статистичні тести на випадковість [12, 16]. Ці константи є ірраціональними числами, цифри яких не повторюються, а тому, теоретично, можуть використовуватись для генерації неперіодичних послідовностей ПВЧ.

У роботі [12] досліджувалось також питання використання десяткового представлення деяких раціональних чисел вигляду k/p, де p – велике просте число, для генерації періодичних псевдовипадкових послідовностей. В результаті було запропоновано декілька простих ГПВЧ із надзвичайно великими періодами повторення.

Отримати такі значення раціональних та ірраціональних чисел з необхідною нам точністю можна різноманітними спеціалізованими чисельними методами [2, 12, 15, 17–19]. Деякі методи використовують цілочислові типи даних або типи у форматі з фіксованою комою, а інші – арифметику з рухомою комою. Проте, недоліком багатьох таких методів є низька швидкодія, погана збіжність, обмеженість точності та допустимих діапазонів значень, необхідність зберігання великих обсягів попередньо обчислених даних в пам'яті, використання обчислювально складних операцій, накопичення похибок, необхідність симуляції обчислень з довільною точністю, відсутність апаратної підтримки тощо.

Формулювання мети та завдань дослідження

У попередній статті [2] ми запропонували набір ГПВЧ із хорошими статистичними характеристиками на основі 64-бітних чисел типу *double*. Ці генератори використовують

© Горячий, О. Я., Максимович, В. М., & Шабатура, М. М. (2024). Дослідження множини початкових значень генераторів псевдовипадкових чисел на основі арифметики з рухомою комою. Сучасний захист інформації, 2(58), 91–102. https://doi.org/10.31673/2409-7292.2024.020011.

ефективні алгоритми для швидкої апроксимації елементарних нелінійних функцій $1/x$ та $1/\sqrt{x}$ різної точності. Проте, все ще не досліджено їх поведінку при зміні початкових значень.

Об'єкт дослідження – генератори псевдовипадкових чисел, побудовані на основі чисельних методів апроксимації елементарних функцій в арифметиці з рухомою комою.

Предмет дослідження – методи і засоби визначення періоду повторення послідовностей псевдовипадкових чисел, сформованих генераторами на основі чисельних методів в арифметиці з рухомою комою, оцінки їх статистичних характеристик та визначення прийнятних діапазонів початкових значень відповідних генераторів.

Мета роботи – подальше дослідження запропонованих ГПВЧ на основі чисельних методів в арифметиці з рухомою комою, зокрема, визначення діапазонів можливих початкових значень та вибір генераторів із найкращими характеристиками для всіх початкових значень.

Для досягнення зазначеної мети визначено такі основні **завдання дослідження**: визначити залежність періоду повторення запропонованих ГПВЧ від початкового значення та інших параметрів алгоритму; оцінити статистичні характеристики сформованих ПВБП на основі тестів NIST при різних початкових значеннях; оцінити множину початкових значень при яких ГПВЧ формують послідовності хорошої якості; визначити серед них ГПВЧ із найкращими характеристиками.

Вибір методів дослідження

Для статистичного тестування ГПВЧ використано набір тестів NIST Statistical Test Suite [16, 20]. Цей набір містить 188 статистичних тестів, що спеціально розроблені Національним інститутом стандартів і технології (NIST) для дослідження випадковості бінарних послідовностей відповідно до різних статистик як для окремих бітів, так і бітових блоків [1, 21].

Запропоновані у [2] ГПВЧ на основі чисельних методів в арифметиці з рухомою комою використовують для побудови послідовності вхідних аргументів рекурентне співвідношення

$$x_i = 1,000003174189 \cdot (x_{i-1} + 0,000358474),\ i = 1;2;..., \qquad (1)$$

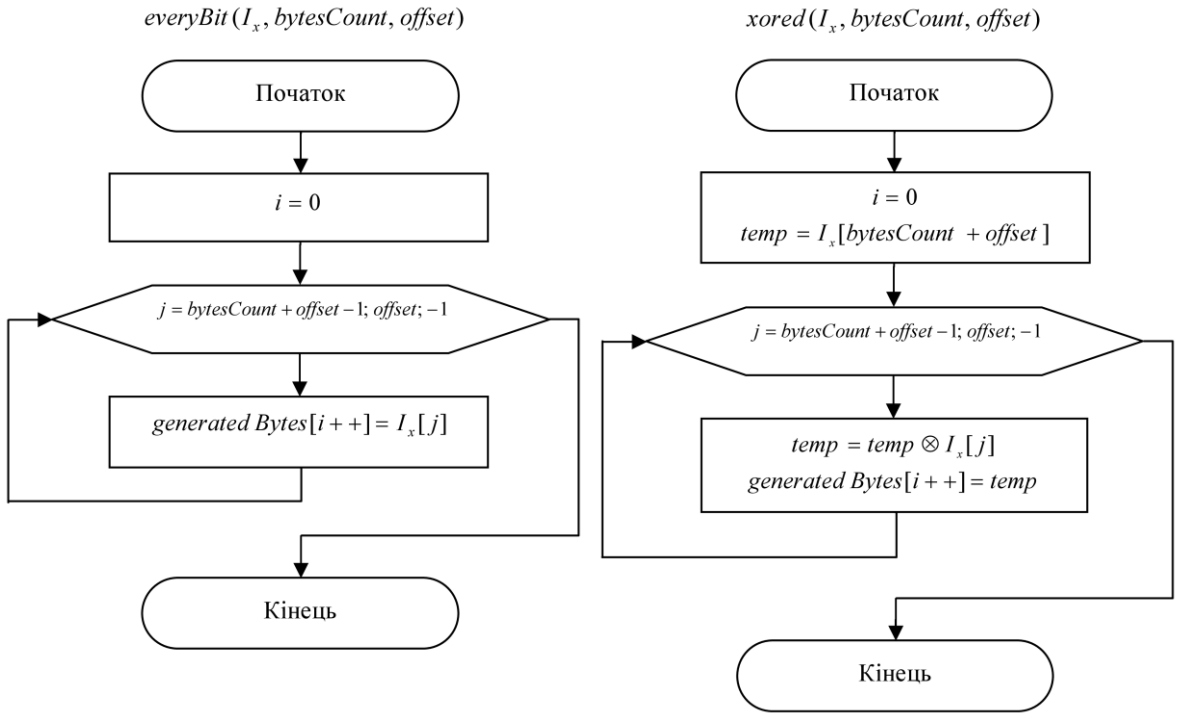
де x_0 – початкове значення ГПВЧ у форматі типу *double*.

Нами було досліджено характеристики таких генераторів при $x_0 = 0$ в залежності від вибраної функції $y(x_i)$, режиму *mode* та параметрів *bytesCount* і *offset*, що використовуються в алгоритмі ГПВЧ для формування вихідної послідовності із послідовності (1). Обчислення більшості елементарних функцій, що розглядалися у [2] в якості $y(x_i)$, базуються на так званому методі магічної константи [22] та модифікованих ітераційних методах для апроксимації оберненого значення $y(x_i) \approx 1/x_i$ [18, 23, 24] та зворотного квадратного кореня $y(x_i) \approx 1/\sqrt{x_i}$ [19, 25, 26]. Покращені алгоритми для наближення таких функцій наведені у [2].

Проаналізувавши отримані результати при фіксованому початковому значенні $x_0 = 0$, ми обрали 8 функцій $y(x_i)$, що показали найкращі статистичні результати згенерованих послідовностей (цифра у назві функції вказує на кількість використаних ітерацій):

1) тотожна функція: x_i;
2) обернене значення: $rcp3(x_i)$;
3) зворотний квадратний корінь:
   - $rsqrt2w(x_i)$, $rsqrt3w(x_i)$ (метод Вальчика та ін. [26]);
   - $rsqrt1h(x_i)$, $rsqrt2h(x_i)$ (на основі методу Хаусхолдера);
   - $rsqrt2dc(x_i)$, $rsqrt3dc(x_i)$ (метод переключення констант [19]).

Ми розглядали режими $mode \in \{everyBit, xored\}$ із параметрами $bytesCount \in \{1, 2, 3\}$ згенерованих байт ПВБП за один виклик функції $y(x_i)$ і $offset \in \{0, 1\}$ байт зміщення, що задовольняють умову $bytesCount + offset \le 3$. Блок-схеми вказаних режимів генерації ПВЧ наведені на рис. 1, де I_x – 8-байтове ціле число, що є бітовим представленням значення $y(x_i)$ з рухомою комою. Запропоновані ГПВЧ позначимо у вигляді: $< y >\_< mode >$ із параметрами $< bytesCount >/< offset >$. Як показано у [2], у випадку $x_0 = 0$, найкращі результати отримано при наступних значеннях параметрів: $bytesCount = 3$ та $offset = 0$ (3/0).



а б

Рис. 1. Блок-схеми, що відображають режими генерації ПВЧ (байт):
а – режим *everyBit*; б – режим *xored*

Із всіх можливих невід'ємних 64-бітних початкових значень типу *double*, $x_0 \ge 0$, для тестування візьмемо 34 довільних чисел з рухомою комою, що знаходяться у проміжку $x_0 \in [0; 1,7977 \cdot 10^{308})$. Обрані значення мають різну кількість одиниць у бітовому представленні. Зокрема, спеціальне значення $x_0 = 0$ містить всі нулі, денормалізоване число $x_0 = 2,0 \cdot 10^{-308}$ має нульові біти знаку та експоненти, нормалізоване значення $x_0 = 256$, що є степенем двійки, – нульові біти знаку та мантиси, інші нормалізовані числа, наприклад $x_0 = 17,17$, – певну кількість одиниць в експоненті та в мантисі. Крім того, 24 нормалізованих значень знаходяться у проміжку $x_0 \in (10^{-10}; 10^{10})$, а 8 – за його межами.

Для визначення періоду повторення, P, ГПВЧ з використанням мови C++ було сформовано послідовності розміру 2 Гб. Під час статистичного тестування використовувались послідовності розміру $1,25 \cdot 10^8$ байт із наступними параметрами тестів NIST: $m = 1000$ підпослідовностей по $n = 1,25 \cdot 10^5$ байт кожна. Крім того, було досліджено результати статистичних тестів NIST, графічних тестів та побудови гістограми при довжині послідовності, що рівна періоду P. Для цього параметр m для тестів NIST визначається згідно із формулою

$$m = \left\lfloor P / (1,25 \cdot 10^5) \right\rfloor, \quad (2)$$

де період P визначений у байтах. Слід зазначити, що NIST рекомендує для тестування використовувати $m \geq 1000$ [21].

Тестування періоду повторення ГПВЧ

Період повторення P сформованих ПВБП залежить від значення x_0, параметру генерації *bytesCount* та виду функції. Це показано на рис. 2 на прикладі генератора *rsqrt2dc_everyBit* для випадку параметрів генерації 3/0. Тут пунктирною лінією позначено мінімальний розмір ПВБП – $1,25 \cdot 10^8$ байт, – що необхідний для статистичного тестування NIST. Як бачимо, у цьому випадку максимальне значення періоду рівне $P = 6,6637 \cdot 10^8$ байт для початкових значень, близьких до $x_0 = 0$. Слід зазначити, що у випадку функції обчислення оберненого значення, наприклад *rcp3*, значення періоду дещо відрізняється, зокрема для $x_0 = 0$ період ГПВЧ рівний $P = 6,6604 \cdot 10^8$. Таким чином, для того щоб одержати неперіодичну ПВБП розміру як мінімум $1,25 \cdot 10^8$ байт потрібно обирати початкове значення x_0 із діапазону $x_0 \in [0; 10^{250}]$ (близько $2^{62,86}$ можливих значень, разом із денормалізованими числами). Проте, сформовані ПВБП можуть частково або повністю повторюватись при різних значеннях x_0, зокрема якщо початкові значення близькі.

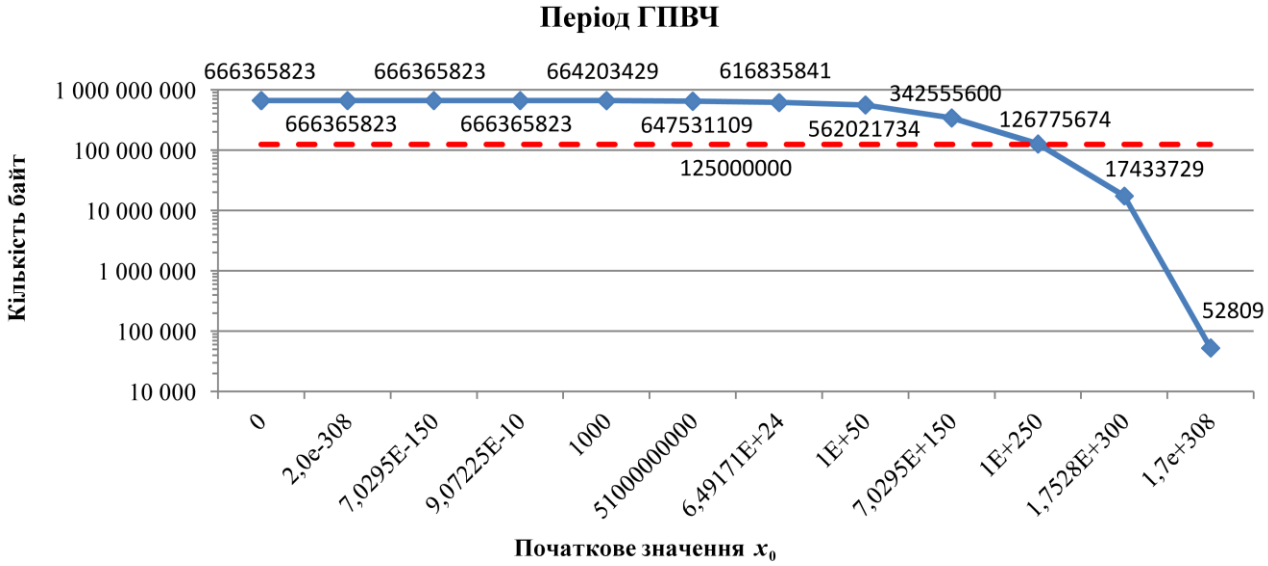


Рис. 2. Залежність періоду повторення ГПВЧ *rsqrt2dc_everyBit* від початкового значення для *bytesCount*/*offset* = 3/0

При модифікації параметра *bytesCount* період повторення генераторів змінюється прямо пропорційно, що підтверджується експериментальними дослідженнями. У цьому випадку, для того, щоб період ГПВЧ був більший за рекомендований мінімальний розмір послідовності необхідно при *bytesCount* = 1 вибирати початкове значення x_0, не більше за 10^{135}, а при значенні параметра *bytesCount* = 2 – не більше за 10^{220}.

Статистичне тестування сформованих ПВБП за методологією NIST при зміні початкових значень ГПВЧ

Для прикладу наведемо на рис. 3 типові статистичні портрети тестів NIST генератора *rsqrt2dc_everyBit* для деяких початкових значень x_0 при $m = 1000$. Висувається припущення

H_0 про випадковість бітової послідовності, що тестується. Тест вважається пройденим, якщо імовірність P проходження тесту NIST – частка підпослідовностей із 1000 досліджуваних, значення *P-value* яких не менше за обраний рівень значущості $\alpha = 0{,}01$ – потрапляє у визначений довірчий інтервал. Даний інтервал залежить від рівня значущості α та розміру вибірки m. У випадку стандартної методики оцінювання NIST Statistical Test Suite із [16], що використовується також в даній статті, довірчий інтервал різних ПВБП буде відрізнятися для тестів Random Excursions (тести № 160–167) та Random Excursions Variant (тести № 168–185). На рис. 3 межі визначеного довірчого інтервалу позначено пунктирною лінією. Суцільною лінією позначено альтернативну методику визначення довірчого інтервалу згідно із стратегією 1 [16, с. 90]. Можна побачити, що у випадку $x_0 = 0$ не пройдено один тест – Non-Overlapping Template (рис. 3а), а у випадку $x_0 = 2108{,}573374$ – тест Block Frequency (рис. 3г). Крім того, для початкового значення $x_0 = 2108{,}573374$ декілька тестів знаходяться на межі довірчого інтервалу або близько до нього – тести Non-Overlapping Template та Random Excursions Variant (рис. 3г).

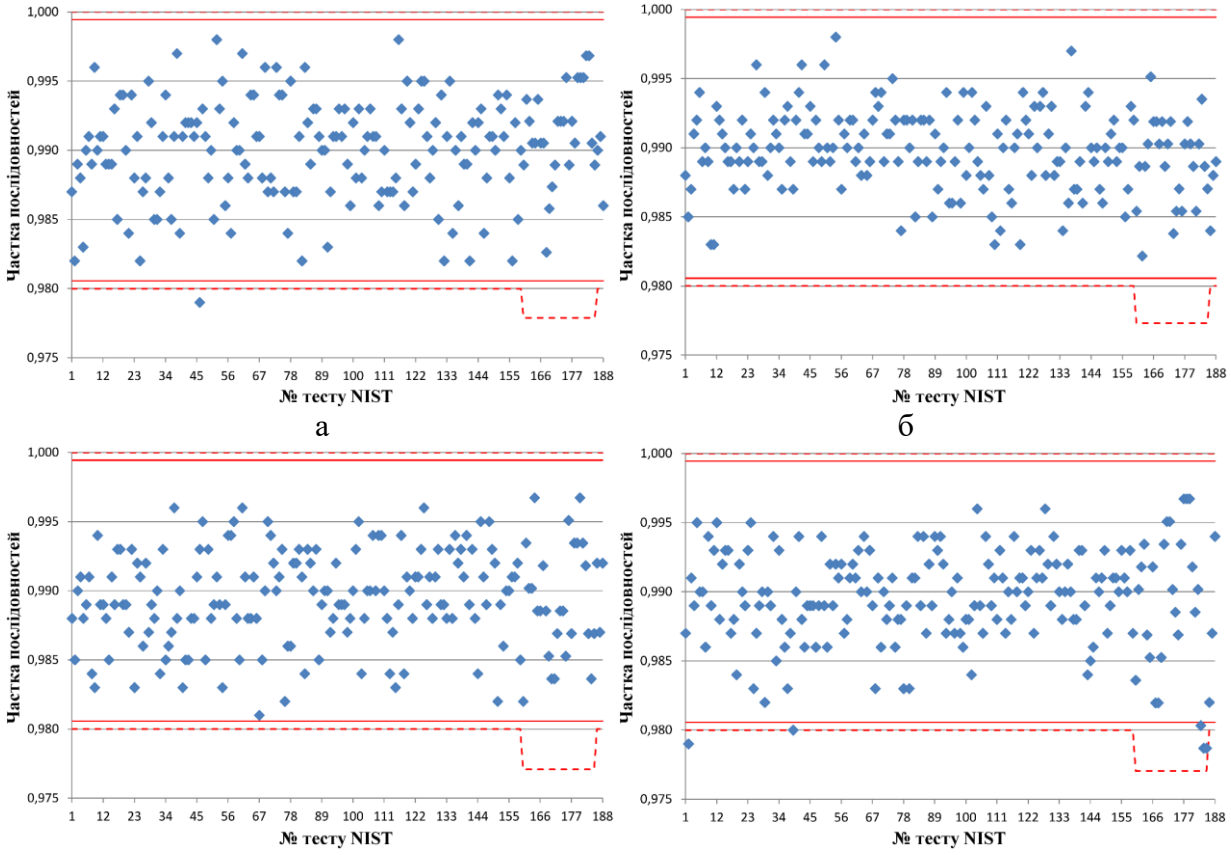


Рис. 3. Статистичні портрети ГПВЧ *rsqrt2dc_everyBit* (3/0) для початкових значень:
а – $x_0 = 0$; б – $x_0 = 9{,}07225 \cdot 10^{-10}$; в – $x_0 = 17{,}17$; г – $x_0 = 2108{,}573374$

Одержані результати статистичного тестування сформованих послідовностей для вибраних початкових значень ($x_0 = 0$ та з набору із 24 значень, $x_0 \in (10^{-10}; 10^{10})$) у випадку значень параметрів *bytesCount* і *offset* рівних 3/0 наведені у табл. 1. Тут через "*e*" позначено режим *everyBit*, а через "*x*" – режим *xored*.

На основі аналізу усереднених результатів кількості пройдених тестів NIST табл. 1 для 25 різних початкових значень виконаємо порівняння наведених генераторів (див. рис. 4) в

залежності від функції та використаного режиму. Як бачимо, найбільшу кількість згенерованих ПВБП, що повністю пройшли усі тести NIST, 21 (84%), має генератор *rsqrt2dc_everyBit*. Ще три генератори – *rcp3_everyBit*, *rsqrt3w_xored* та *rsqrt3dc_everyBit* – мають по 20 (80%) успішних проходжень тестів NIST. За результатами аналізу, середня кількість пройдених тестів для різних функцій є приблизно однаковою як для режиму *everyBit*, так і для режиму *xored*. Для подальших досліджень ми обрали 4 ГПВЧ: по два кращих для режимів *everyBit* і *xored*.

Таблиця 1

Результати статистичного тестування згенерованих ПВБП (*bytesCount* = 3, *offset* = 0).
Кількість пройдених тестів NIST

Початкове значення x_0	**Функція**															
	rcp3		***rsqrt2h***		***rsqrt2w***		***rsqrt3w***		***x***		***rsqrt2dc***		***rsqrt3dc***		***rsqrt1h***	
	Режим		**Режим**		**Режим**		**Режим**		**Режим**		**Режим**		**Режим**		**Режим**	
	e	***x***	***e***	***x***	***e***	***x***	***e***	***x***	***e***	***x***	***e***	***x***	***e***	***x***	***e***	***x***
0	188	188	188	188	188	188	188	188	188	188	**187**	188	188	**186**	**187**	188
$9,07225\cdot10^{-10}$	188	188	188	188	**187**	188	**187**	188	**186**	**187**	188	188	188	**187**	188	**187**
$1,5\cdot10^{-8}$	**187**	**187**	188	188	**187**	**186**	188	188	**187**	**187**	188	**187**	188	**185**	**186**	**187**
$7,02951\cdot10^{-5}$	188	188	188	**187**	188	188	188	188	188	188	188	**185**	188	188	**184**	**187**
0,000175	**187**	**187**	**186**	188	188	188	**186**	188	188	**187**	188	**187**	**187**	188	188	188
0,005745893	188	188	**187**	188	188	**187**	188	**187**	188	188	188	**185**	**187**	188	188	188
0,0123	**187**	**187**	**185**	188	188	188	**187**	188	188	188	188	188	188	188	188	188
0,1	188	**187**	188	**187**	**187**	188	188	188	188	188	188	**186**	**187**	188	188	188
0,247323326	**186**	**186**	**187**	188	188	**187**	188	188	**187**	**187**	188	188	188	**187**	**187**	188
0,5	188	**187**	**187**	**187**	188	**186**	188	**187**	**187**	188	188	**187**	188	188	188	188
0,82822999	188	188	**187**	**187**	**186**	**187**	188	188	188	188	188	**187**	188	**187**	**187**	**187**
1	188	188	188	188	188	188	**187**	**187**	188	**186**	188	**187**	188	188	**187**	188
1,5	**187**	188	188	188	188	188	188	188	188	**187**	188	188	**187**	188	188	188
1,635259593	188	188	188	**187**	188	188	188	188	**187**	188	188	**187**	188	**185**	188	188
5	188	188	188	188	188	**187**	188	188	188	188	188	**187**	188	**187**	188	188
8,257638398	188	**186**	**187**	188	188	188	**186**	188	188	188	**187**	**187**	188	188	188	188
17,17	188	188	188	188	188	**187**	188	188	**187**	188	188	188	188	188	**187**	188
39,60029254	188	188	188	**187**	**187**	188	188	**186**	**186**	**186**	188	188	188	188	**186**	188
44	188	188	188	**186**	188	188	**187**	188	188	188	188	188	188	188	188	188
146,2356369	188	**187**	188	188	188	**187**	188	**187**	**186**	188	188	188	188	188	188	188
256	188	**186**	188	**186**	**187**	**186**	**187**	188	**185**	188	188	188	**186**	**187**	**187**	**186**
1000	188	188	**186**	188	188	188	188	188	**187**	188	**187**	**187**	188	188	**187**	188
2108,573374	188	**186**	**185**	188	**187**	188	**187**	188	188	188	**187**	188	188	188	**187**	188
12300	188	188	188	**186**	**187**	**187**	188	188	**186**	188	188	188	188	**187**	188	188
704765,5961	188	188	188	188	**187**	188	**185**	188	188	188	188	188	188	188	**187**	**187**
Всього пройдено																
188 тестів	**80%**	60%	64%	64%	64%	60%	64%	**80%**	56%	72%	**84%**	52%	**80%**	64%	52%	76%
187 тестів	16%	24%	20%	24%	32%	28%	24%	16%	24%	20%	16%	36%	16%	24%	36%	20%
186 тестів	4%	16%	8%	12%	4%	12%	8%	4%	16%	8%	-	4%	4%	4%	8%	4%
< 186 тестів	-	-	8%	-	-	-	4%	-	4%	-	-	8%	-	8%	4%	-

Аналогічним чином можна дослідити результати статистичного тестування NIST генераторів у випадку значень параметрів *bytesCount* і *offset*, відмінних від 3/0. Узагальнені статистичні результати вибраних генераторів показані на рис. 5. Розширивши діапазон

початкових значень x_0, найкращі результати статистичного тестування показали генератори *rsqrt2dc_everyBit* (3/0), *rsqrt2dc_everyBit* (1/0), *rsqrt3w_xored* (3/0) та *rcp3_everyBit* (3/0).

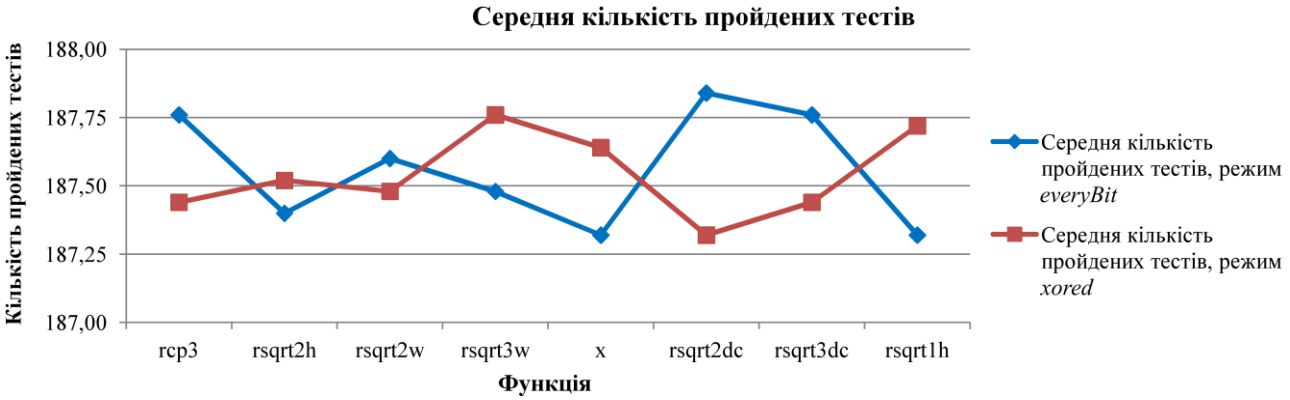


Рис. 4. Середня кількість пройдених тестів NIST для досліджуваних ГПВЧ

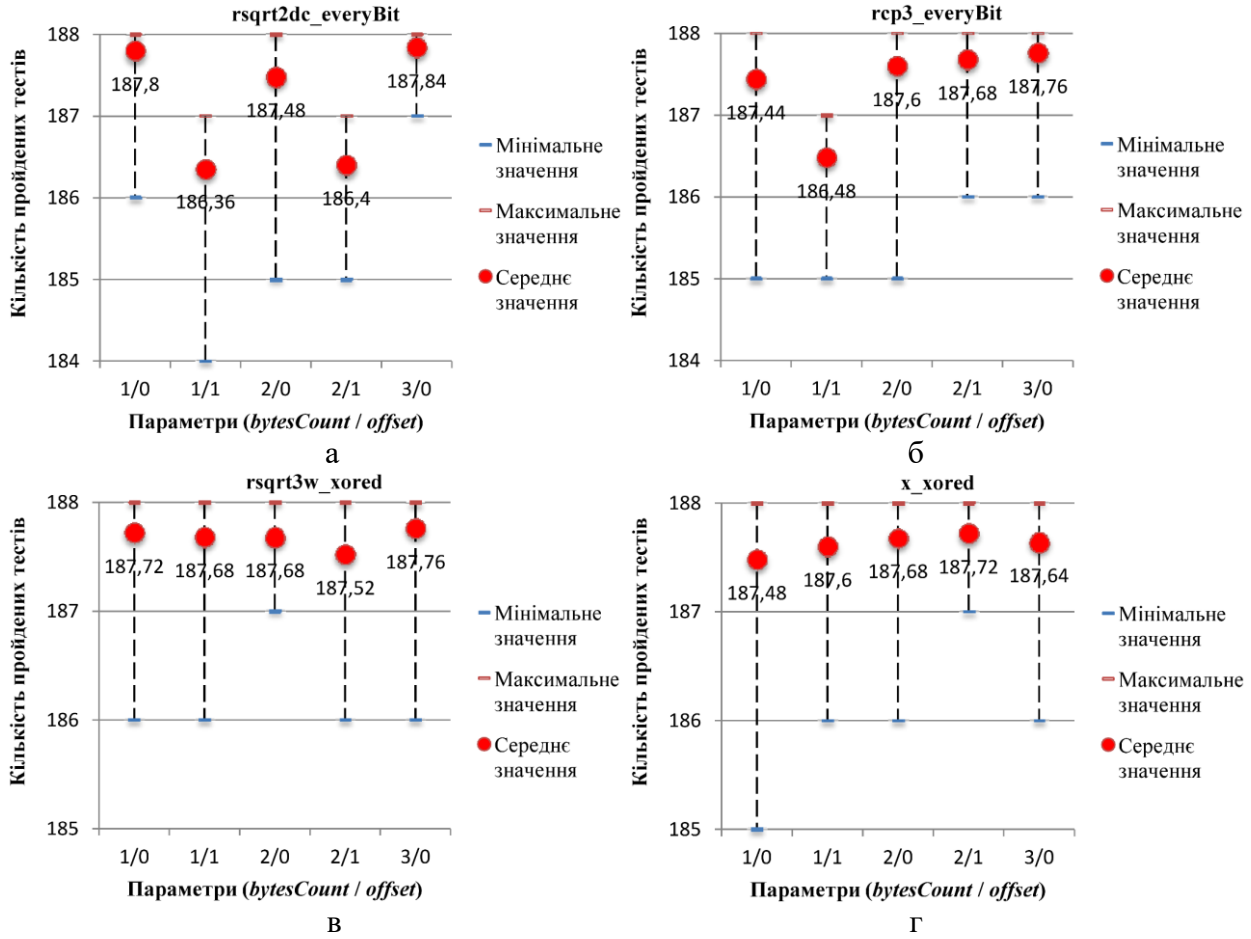
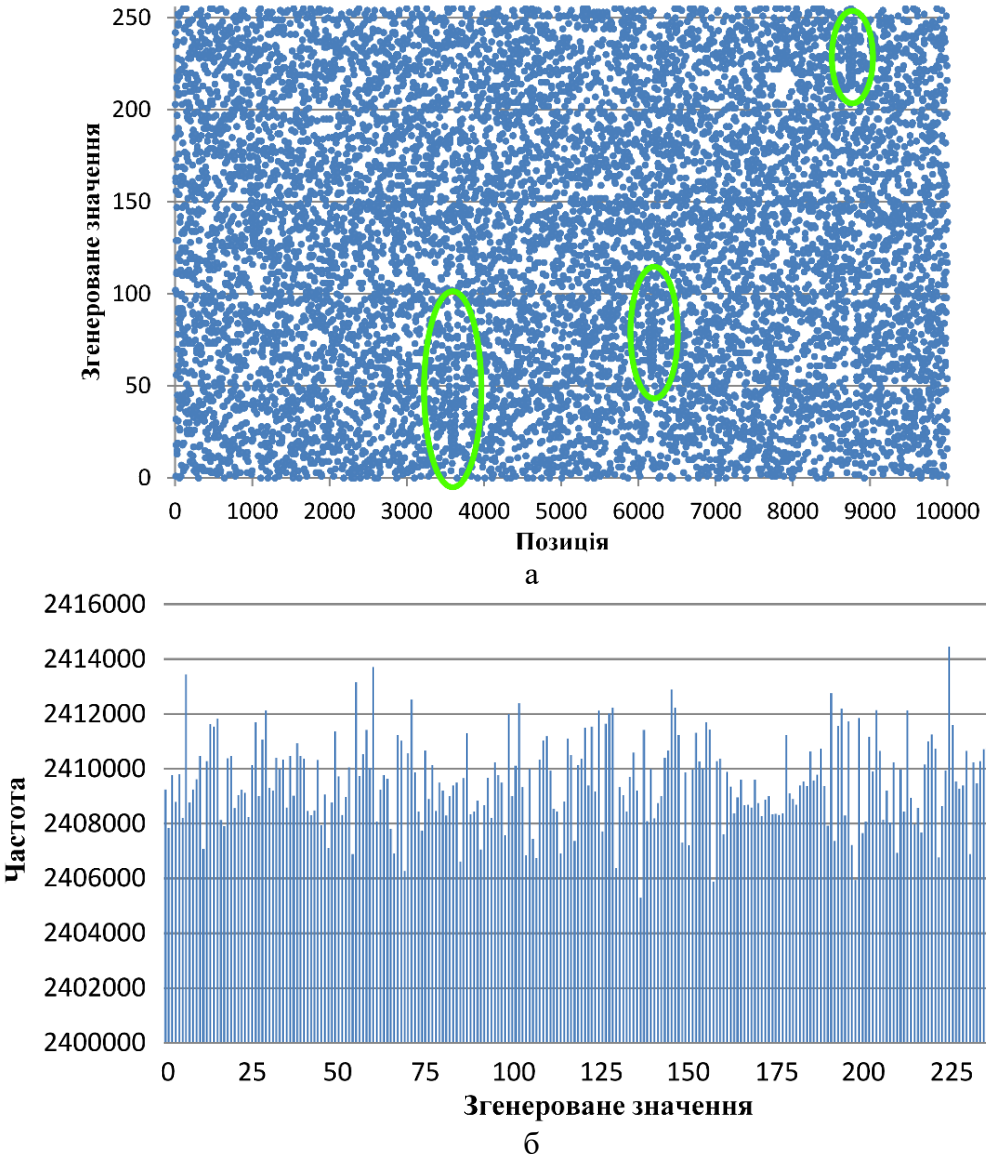


Рис. 5. Середня кількість пройдених тестів NIST в залежності від значень параметрів:
а – *rsqrt2dc_everyBit*; б – *rcp3_everyBit*; в – *rsqrt3w_xored*; г – *x_xored*

Детальний аналіз згенерованих послідовностей показав, що для будь-якого початкового значення x_0, що не перевищує $8{,}131516\cdot10^{-20}$, генератори формують такі самі ПВБП, як і для $x_0 = 0$. Тому, зважаючи на великий ступінь подібності сформованих послідовностей для сусідніх дискретних значень x_0, особливо для початкових значень близьких до $x_0 = 0$ та у випадку $bytesCount > 1$, ми рекомендуємо використовувати в якості початкових значень

$x_0 \geq 0,5$ та не обирати значення x_0, що є близькими один до одного. Така умова скоротить множину вибору початкових значень до $2^{60,81}$, $2^{61,52}$ та $2^{61,7}$ елементів для *bytesCount* = 1, *bytesCount* = 2 та *bytesCount* = 3, відповідно.

Сформовані послідовності ПВЧ (байтів) в межах визначеного періоду повторення P можна представити графічно та побудувати їх гістограми. В цьому випадку розмір послідовностей рівний періоду ГПВЧ. Такий підхід дозволяє наочно переконатися у рівномірності розподілу отриманих ПВЧ. Аналіз отриманих результатів показав, що послідовності, сформовані ГПВЧ *rsqrt2dc_everyBit* (3/0) містять певний ледве помітний патерн, що вказує на недолік такого генератора (рис. 6).



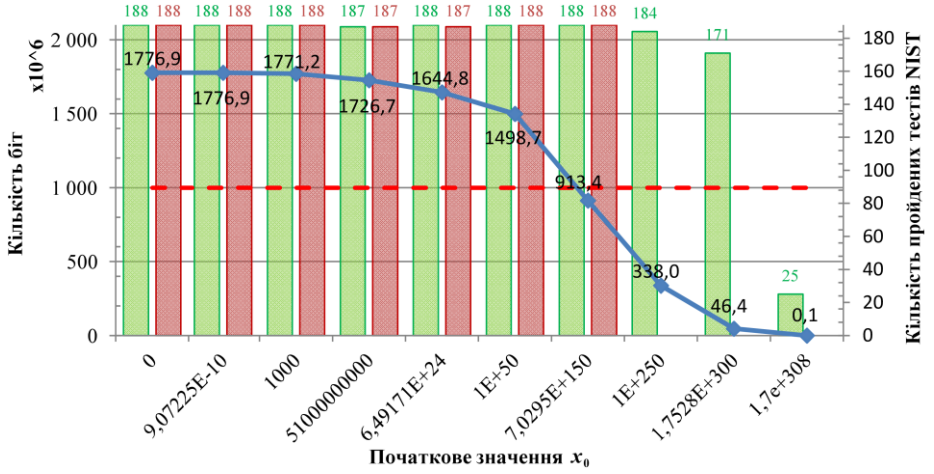
а

б

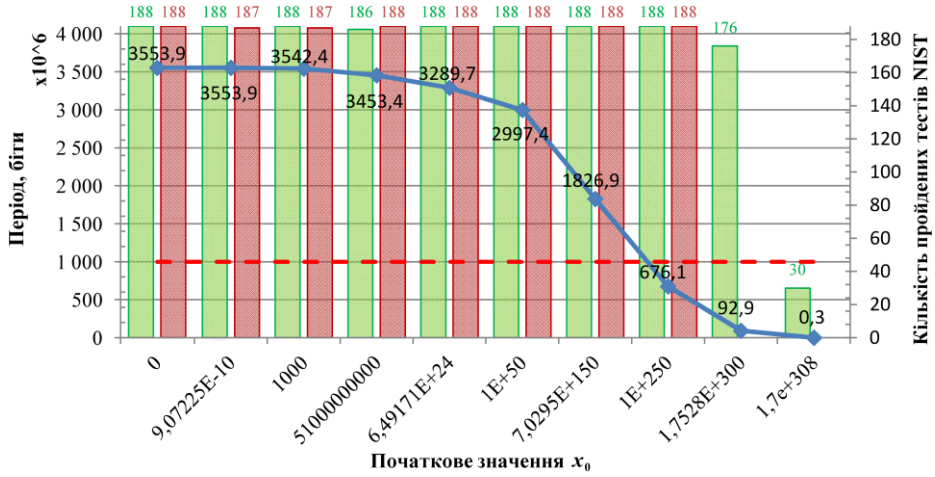
Рис. 6. Результати графічного тестування неперіодичної послідовності ПВЧ, сформованої ГПВЧ *rsqrt2dc_everyBit* (3/0) при $x_0 = 6,491708 \cdot 10^{24}$:
а – початок послідовності; б – гістограма послідовності

Аналогічним чином при тестуванні NIST врахуємо період сформованих послідовностей, використавши формулу (2) для визначення параметра m. Такий підхід дозволить точніше оцінити статистичні характеристики вказаних ГПВЧ, проте, зазвичай вимагає більше часу для тестування. Результати показують, що більшість досліджених генераторів погано проходять

таке статистичне тестування при параметрах *bytesCount* = 3 та *offset* = 0 (3/0). Інші генератори, зокрема *rsqrt2dc_everyBit* (1/0), *rsqrt3w_xored* (2/0) та *x_xored* (2/1), показують непогані результати тестів NIST із вибором параметра m (рис. 7), на тому ж рівні, що й при $m = 1000$.



а



б

Рис. 7. Залежність періоду повторення та результатів статистичного тестування NIST (зелений стовпець – при фіксованому значенні параметра m, $m = 1000$; червоний стовпець – підбір параметра m під величину періоду) від початкового значення x_0:
а – *rsqrt2dc_everyBit* (1/0); б – *rsqrt3w_xored* (2/0)

Обговорення отриманих результатів

Врахувавши всі одержані результати, найкращі статистичні характеристики має ГПВЧ *rsqrt2dc_everyBit* (1/0). Початкове значення для цього генератора, що є невід'ємним числом із рухомою комою типу *double*, слід обирати із проміжку $x_0 \in [0,5; 10^{135})$. Доля проходження тестів NIST цього генератора складає 82,14%, а середня кількість пройдених тестів рівна 187,79. Проте, його недолік – відносно невеликий період повторення. Генератори *rsqrt2dc_everyBit* (3/0) та *rsqrt3w_xored* (3/0), для яких успішно проходять 83,33% (в середньому 187,8 тестів) та 76,67% (в середньому 187,7 тестів) тестів NIST, мають втричі більший період, проте володіють певними статистичними недоліками. Зокрема, вони гірше проходять тестування NIST для послідовностей, довжина яких рівна періоду ГПВЧ, та мають вищий ступінь кореляції згенерованих послідовностей для близьких початкових значень.

© Горячий, О. Я., Максимович, В. М., & Шабатура, М. М. (2024). Дослідження множини початкових значень генераторів псевдовипадкових чисел на основі арифметики з рухомою комою. Сучасний захист інформації, 2(58), 91–102. https://doi.org/10.31673/2409-7292.2024.020011.

Висновки

У цій статті ми дослідили статистичні характеристики та період повторення послідовностей, сформованих генераторами ПВЧ на основі чисельних методів апроксимації елементарних функцій в арифметиці з рухомою комою в залежності від початкового значення ГПВЧ, параметрів *bytesCount* і *offset*, використаної функції та режиму роботи генератора. Для дослідження статистичних характеристик сформованих послідовностей використовувалась методологія NIST, графічні тести та гістограми. На основі одержаних результатів було запропоновано діапазони для вибору початкових значень ГПВЧ та визначено генератори, що мають найкращі результати тестування сформованих послідовностей.

У подальшому планується:

дослідити взаємну кореляцію для суміжних початкових значень та визначити безпечний інтервал їх зміни;

визначити швидкодію запропонованих ГПВЧ на основі чисельних методів в арифметиці з рухомою комою;

дослідити криптостійкість запропонованих алгоритмів генерації ПВЧ;

дослідити методи вдосконалення запропонованих ГПВЧ для покращення їх характеристик та усунення наявних недоліків.

Перелік посилань

1. Kietzmann, P. A guideline on pseudorandom number generation (PRNG) in the IoT [Electronic resource] / P. Kietzmann, T.C. Schmidt, M. Wahlisch // ACM computing surveys. – 2021. – Vol. 54, no. 6. – P. 1–38. – Mode of access: https://doi.org/10.1145/3453159.

2. Генератори псевдовипадкових бітових послідовностей на основі чисельних методів в арифметиці з рухомою комою [Електронний ресурс] / В. Максимович [та ін.] // Сучасна спеціальна техніка. – 2021. – Т. 64, № 1. – С. 81–92. – Режим доступу: https://doi.org/10.36486/mst2411-3816.2021.1(64).7.

3. Matsumoto, M. Pseudorandom number generation: impossibility and compromise [Electronic resource] / M. Matsumoto, M. Saito, H. Haramoto // Journal of universal computer science. – 2006. – Vol. 12, no. 6. – P. 672–690. – Mode of access: https://doi.org/10.3217/jucs-012-06-0672.

4. Rani, S. Steganography on digital color image using modulo function and pseudo-random number generator [Electronic resource] / S. Rani, A. Kurniawardhani, Y. A. W. Rendani // International journal on advanced science engineering and information technology. – 2021. – Vol. 11, no. 6. – 2470. – Mode of access: https://doi.org/10.18517/ijaseit.11.6.12687.

5. Kordov, K. Steganography in color images with random order of pixel selection and encrypted text message embedding [Electronic resource] / K. Kordov, S. Zhelezov // PeerJ computer science. – 2021. – Vol. 7, no. 11. – e380. – Mode of access: https://doi.org/10.7717/peerj-cs.380.

6. Wang, L. Pseudo-random number generator based on logistic chaotic system [Electronic resource] / L. Wang, H. Cheng // Entropy. – 2019. – Vol. 21, no. 10. – 960. – Mode of access: https://doi.org/10.3390/e21100960.

7. Datcu, O. Chaos based cryptographic pseudo-random number generator template with dynamic state change [Electronic resource] / O. Datcu, C. Macovei, R. Hobincu // Applied sciences. – 2020. – Vol. 10, no. 2. – 451. – Mode of access: https://doi.org/10.3390/app10020451.

8. Demchik, V. Pseudo-random number generators for Monte Carlo simulations on graphics processing units [Electronic resource] / V. Demchik. – Ithaca, NY, USA : ArXiv, 2010. – 31 p. – (Preprint / Cornell University ; 1003.1898v1). – Mode of access: https://arxiv.org/pdf/1003.1898v1 (date of access: 10.05.2024).

9. IEEE 754-2019. Standard for floating-point arithmetic [Electronic resource]. – Replaces IEEE 754-2008 ; effective from 2019-07-22. – Official edition. – 2019. – 84 p. – Mode of access: https://doi.org/10.1109/IEEESTD.2019.8766229.

10. Cotrina, G. Gaussian pseudorandom number generator using linear feedback shift registers in extended fields [Electronic resource] / G. Cotrina, A. Peinado, A. Ortiz // Mathematics. – 2021. – Vol. 9, no. 5. – 556. – Mode of access: https://doi.org/10.3390/math9050556.

11. Harase, S. Implementing 64-bit maximally equidistributed F2-linear generators with Mersenne prime period [Electronic resource] / S. Harase, T. Kimoto // ACM transactions on mathematical software. – 2018. – Vol. 44, no. 3. – 30. – Mode of access: https://doi.org/10.1145/3159444.

12. Marsaglia, G. On the randomness of pi and other decimal expansions [Electronic resource] / G. Marsaglia // InterStat: statistics on the Internet. – 2005. – P. 1–17. – Mode of access: http://yaroslavvb.com/papers/marsaglia-on.pdf (date of access: 23.05.2024).

13. Shrestha, B. Multiprime Blum-Blum-Shub pseudorandom number generator [Electronic resource] : Master's thesis in Applied Mathematics / Shrestha B. – Monterey, CA, USA, 2016. – 83 p. – Mode of access: https://core.ac.uk/download/pdf/81222279.pdf (date of access: 27.09.2021).

14. A self-perturbed pseudo-random sequence generator based on hyperchaos [Electronic resource] / Y. Zhao [et al.] // Chaos, solitons & fractals: X. – 2019. – Vol. 4. – 100023. – Mode of access: https://doi.org/10.1016/j.csfx.2020.100023.

15. Bailey, D. Pi day is upon us again and we still do not know if pi is normal [Electronic resource] / D. Bailey, J. Borwein // The american mathematical monthly. – 2014. – Vol. 121, no. 3. – P. 191–206. – Mode of access: https://doi.org/10.4169/amer.math.monthly.121.03.191.

16. A statistical test suite for random and pseudorandom number generators for cryptographic applications [Electronic resource]. – Gaithersburg, MD, USA : National Institute of Standards and Technology, 2010. – 131 p. – Mode of access: https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-22r1a.pdf (date of access: 24.05.2024).

17. Горпенюк, А. Генератор псевдовипадкових чисел на обчислювачі кореня квадратного з простого числа [Електронний ресурс] / А. Горпенюк, Н. Лужецька // Вісник НУ Львівська політехніка "Автоматика, вимірювання та керування". – 2013. – № 753. – С. 45–50. – Режим доступу: https://science.lpnu.ua/uk/node/3665 (дата звернення: 20.03.2024).

18. Huang, K. Improving performance of floating point division on GPU and MIC [Electronic resource] / K. Huang, Y. Chen // Algorithms and architectures for parallel processing : proc. 15th int. conf. (ICA3PP 2015), Zhangjiajie, 18–20 November 2015. – Cham, 2015. – P. 691–703. – Mode of access: https://doi.org/10.1007/978-3-319-27122-4_48.

19. Moroz, L. Modified fast inverse square root and square root approximation algorithms: the method of switching magic constants [Electronic resource] / L. Moroz, V. Samotyy, O. Horyachyy // Computation. – 2021. – Vol. 9, no. 2. – 21. – Mode of access: https://doi.org/10.3390/computation9020021.

20. The NIST statistical test suite [Electronic resource]. – Random Bit Generation. – Version 2.1.1. – Gaithersburg, MD, USA : National Institute of Standards and Technology, 2014. – Mode of access: https://csrc.nist.gov/projects/random-bit-generation/documentation-and-software (date of access: 25.05.2024).

21. On the interpretation of results from the NIST statistical test suite [Electronic resource] / M. Sýs [et al.] // Romanian journal of information science and technology. – 2015. – Vol. 18, no. 1. – P. 18–32. – Mode of access: https://romjist.ro/content/pdf/02-msys.pdf (date of access: 25.05.2024).

22. Robertson, M. A brief history of InvSqrt [Electronic resource] : Bachelor's thesis in Computer Science / Robertson M. – New Brunswick, Canada, 2012. – 48 p. – Mode of access: https://mrober.io/papers/rsqrt.pdf (date of access: 12.03.2024).

23. Мороз, Л. Швидке обчислення функції y=1/x з використанням магічної константи [Електронний ресурс] / Л. Мороз, А. Гринчишин // Вісник НУ Львівська політехніка "Автоматика, вимірювання та керування". – 2015. – № 821. – С. 23–29. – Режим доступу: http://nbuv.gov.ua/UJRN/VNULP_2015_821_6 (дата звернення: 25.05.2024).

24. Moroz, L. Efficient floating-point division for digital signal processing application [DSP tips and tricks] [Electronic resource] / L. Moroz, V. Samotyy // IEEE signal processing magazine. – 2019. – Vol. 36, no. 1. – P. 159–163. – Mode of access: https://doi.org/10.1109/MSP.2018.2875977.

25. Lemaitre, F. Cholesky factorization on SIMD multi-core architectures [Electronic resource] / F. Lemaitre, B. Couturier, L. Lacassagne // Journal of systems architecture. – 2017. – No. 79. – P. 1–15. – Mode of access: https://hal.science/hal-01550129/document (date of access: 18.08.2021).

26. Walczyk, C. Improving the accuracy of the fast inverse square root by modifying Newton-Raphson corrections [Electronic resource] / C. Walczyk, L. Moroz, J. Cieśliński // Entropy. – 2021. – Vol. 23, no. 1. – 86. – Mode of access: https://doi.org/10.3390/e23010086.

Надійшла 24.05.2024