УДК 519.212.2:681.51
DOI https://doi.org/10.32838/2663-5941/2020.4/16

Поперешняк С.В.
Київський національний університет імені Тараса Шевченко

ЗАСІБ ДЛЯ ТЕСТУВАННЯ БІТОВОЇ ПОСЛІДОВНОСТІ НА ВИПАДКОВІСТЬ

У статті наведено короткий огляд відомих методів тестування послідовностей на випадковості, а також статичних тестів, розроблених протягом останніх десятиліть, які широко використовуються у криптографічних системах захисту інформації для виготовлення ключової та додаткової інформації. Розглянуто перспективний напрям дослідження – статичне тестування послідовностей із використанням багатовимірної статистики. У роботі наведені формули для тестування випадкових бітових послідовностей на випадковості з використанням двовимірної або тривимірної статистика, яка може бути застосована для тестування коротких і середніх послідовностей.

Підхід до тестування з використанням багатовимірної статистики дає змогу покладатися на глибше обґрунтування випадковості згенерованих послідовностей. Перевагою тестування за допомогою багатовимірних статистичних даних є також можливість високого ступеня паралелізму в обчисленні сімейства статистики. Ця область є перспективною для досліджень, особливо для послідовностей короткої та середньої довжини, де одновимірні статистичні дані часто безсилі.

У роботі наведено методику тестування псевдовипадкової послідовності та отримано явний вид спільного розподілу чисел 2-ланцюжків і чисел 3-ланцюжків різних варіантів у випадковій бітовій послідовності заданої невеликої довжини.

Для реалізації запропонованої методики було розроблено програмний засіб для тестування бітової послідовності на випадковість. Цей засіб включає тести NIST, а також тести з використанням багатовимірної статистики, які добре себе зарекомендували під час тестування псевдовипадкової послідовності невеликої довжини. У результаті подальшого розвитку буде розширена та створена єдина інформаційна система, яка дасть змогу проаналізувати бітову послідовність, використовуючи відомі тести або/та тести з використанням багатовимірної статистики, та вибирати якісну псевдовипадкову послідовність для використання в тій чи іншій предметній області.

Ключові слова: *програмний засіб, бітова послідовність, тестування, багатовимірні статистики; випадкові послідовності; псевдовипадкова послідовність; статистичне тестування.*

Постановка проблеми. Використання випадкових чисел завжди мало своє місце в діяльності людини, але набуло поширення завдяки розвитку інформаційних технологій. Випадкові числа лежать в основі таких напрямів, як криптографія, програмування, симуляція, ігрові системи тощо. Завданням що покладають на випадкові числа, найчастіше є забезпечення незалежності однієї процедури в системі від іншої. Таким чином можна забезпечити непередбачуваність криптографічних ключів, механізми випадковості в комп'ютерних іграх та оптимізацію математичних та програмних методів, наприклад, метод Монте-Карло.

Випадковість є частиною реального світу, яку ми можемо спостерігати в будь-який момент часу, але для роботи реальної системи необхідно отримати випадкові числа штучним чином і в необхідних обсягах.

Якість випадкової послідовності, або «дійсна випадковість», є її ключовою характеристикою, адже в більшості областей застосування саме від цього фактора залежить корисний ефект. Щоб виміряти випадковість, використовують набори відповідних статистичних тестів. Їх результати вказують на характеристики, за допомогою яких можна зробити висновки про випадковість: лінійна залежність між частинами послідовності, періодичні властивості, здатність до компресії, тощо.

У таблиці 1 наведено найвідоміші набори тестів для перевірки бітових послідовностей. Варто зауважити, що деякі з методів випробувань у наборах збігаються, адже вони всі мають одне математичне підґрунтя.

Розглянуті пакети статистичних тестів мають солідне математичне підґрунтя і готову програмну реалізацію. Будь-який із них можна використати для оцінки послідовності або генератора і мати високий рівень впевненості в якості результатів. Однак в екосистемі статистичних тестів на випадковість можна виділити такі тренди:

Таблиця 1

Огляд найвідоміших тестів для перевірки ГПЧ

NIST Statistical Test Suite	Тести Diehard	TestU01
NIST STS – специфікація та відповідна бібліотека мовою С, що були випущені Інститутом Стандартів та Технологій США. Пакет складається з 15 тестів для аналізу бітових послідовностей, що були згенеровані ГПЧ або АГВЧ. Повний опис тестів доступний в роботі "A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications" [1].	Батарея статистичних тестів, призначена для виміру якості ГПЧ та АГВЧ, що була створена Дж. Марсалі у 1995 році. В основі більшості тестів лежить використання генератора для побудови послідовності відповідно до наданої специфікації і порівняння її характеристик з очікуваними від випадкової. Деякі з наведених випробувань можна виділити в групи за подібністю, а інші являють собою один тест. Більше інформації про тести можна знайти в джерелах [2; 3].	Об'ємна бібліотека тестів мовою С, що включає реалізацію ГПЧ, тести та батареї тестів. Всі випробування, що надаються, поділені на групи відповідно до модулів програми [4]:
− Частотний тест − Частотний тест у блоці − Тест на подібні послідовності − Тест на найбільшу послідовність одиниць у блоці − Тест рангів бінарних матриць − Спектральний тест − Тести на шаблони, що перетинаються/не перетинаються − Універсальний тест Маурера − Тест на лінійну складність − Тест на послідовності та тест на приблизну ентропію − Тест кумулятивних сум − Тест на довільні виключення − Тест на довільні виключення з варіантом	− Тест днів народжень − Тест перестановок довжини 5, що перетинаються − Тести мавп − Тести на підрахунок одиниць − «Тест паркінгу» − Тест на мінімальну відстань − Тест випадкових сфер − Тест на стиснення − Тест сум, що перетинаються − Тест на подібні послідовності − Тест «Крепс»	− smultin − sentrop − snpair − sknuth − smarsa − svaria − swalk − scomp − sspectral − sstring − sspacings

− наявна велика кількість різних тестів та пакетів, що часто підходять до вирішення задачі з різних сторін;

− відсутні чіткі лідери, тобто тести, які можна рекомендувати для вирішення більшості проблем;

− неможливо отримати точний висновок про випадковість послідовності навіть після виконання всіх можливих тестів;

− майже всі окремі пакети та тести мають деякі обмеження або недоліки.

Аналіз останніх досліджень і публікацій. Підбірка 14 тестів «Diehard» Дж. Марсали була першою в комплексному тестуванні генераторів псевдовипадкових послідовностей (ПВП). Підбірка розглядається як одна з найбільш суворих сукупностей тестів, вона реалізована програмно і доступна в інтернеті [5]. Однак підбірка тестів «Diehard» має низку недоліків.

1) відсутні докладний опис тестів і методика трактування результатів [6];

2) параметри тестування жорстко задані. При цьому незалежно від довжини ПВП, що тестується, аналізується тільки певне число байтів [6]. Більш короткі ПВП протестувати неможливо;

3) більшість тестів є наближеними і засновані на результатах випробувань, а не на теоретичних моделях [6];

4) рішення про проходження тесту може приймати тільки одне з двох значень (так/ні).

Підбірка тестів ПВП Д. Кнута використовує сім оригінальних статистик і алгоритмів їх підрахунку. Однак цядобірка має низку недоліків.

1) всі алгоритми зводяться до обчислення статистичних критеріїв, апроксимується тільки розподілом χ 2;

2) відсутні рекомендації про параметри тестування. Некоректний вибір деяких значень може привести до істотної залежності від довжини послідовності, що тестується, а також негативно позначатися на потужності статистичного критерію [6];

3) спірної видається [6] методика оцінки результатів, коли випадковими визнаються послідовності, для яких P-value належить інтервалу (0,1; 0,9). Тобто коли P-value більше, ніж 0,9, результати тестування вважаються занадто ідеальними, щоб вважати числову послідовність випадковою;

4) відсутня оригінальна програмна реалізація запропонованих тестів.

У роботі [7] пропонується набір тестів для попередньої перевірки якості випадкових чисел і послідовностей на основі семи різних статистичних тестів.

М. Кендел і Б. Сміт [8] запропонували використовувати 4 тести із застосуванням критерію χ^2:

1) перевірка частоти різних цифр x_1, x_2, ..., x_N в таблиці (тест частот);

2) перевірка частоти різних двозначних чисел серед пар цифр $x_1\ x_2$, $x_2\ x_3$, $x_3\ x_4$, ..., $x_{N-1}\ x_N$ (тест пар);

3) перевірка частоти різних інтервалів між двома послідовними нулями (тест інтервалів);

4) перевірка частоти різних типів четвірок (aaaa, aaab, aabc, aabb, abcd), а також перевірка частоти різних типів п'ятірок (покертест).

Стандарт NIST STS 800-22 Національного інституту стандартизації і технологій NIST [1; 6] включає 15 тестів і орієнтований на тестування бітових послідовностей, що застосовуються в задачах криптографічного захисту інформації.

Типове застосування тестів (зокрема, Diehard) наводиться, наприклад, у доповіді "The RAND Corporation. A Million Random Digits with 100 000 Normal Deviates" [9].

У разі збільшення довжини ПВП, що тестується (понад 100 тис.), багато статистичних тестів виявляють статистично значущі закономірності, які не виявлялися на вибірках меншого обсягу. Так, наприклад, знаковий ранговий критерій (signed rank test, Вілкоксона), який є досить потужним [8], бракує такі відомі і якісні генератори, як Блюма-Блюма-Шуба (BBS), Шаміра (RSA), «Marsaglia Multicarry» і «Xorshift» Дж. Марсали, вихор Мерсенна (MT19937), а також «істинно випадкову послідовність» [10] вже на 1500–2000 елементів числової послідовності.

Як показав огляд популярних методів тестування бітових послідовностей на випадковість, незважаючи на велику кількість статистичних тестів, вони дають більш коректний результат при досить великому розмірі вибірки. Однак ми не зможемо отримати коректну відповідь із приводу випадковості послідовності, якщо довжина послідовності буде менше 100 елементів.

Розробка пакетів для дослідження випадкових чисел без програмного забезпечення є доволі сумнівною роботою, адже область застосування повністю складається з інформаційних технологій.

Постановка завдання. Метою роботи є покращення наявних та впровадження нових методів тестування бітової послідовності на випадковість. Це включає розробку формального опису статистичних тестів та реалізацію відповідних програмних продуктів.

Досягнення мети включало виконання таких задач:

- огляд математичного підґрунтя та принців тестування послідовностей на випадковість;
- аналіз наявних пакетів тестів;
- формальний опис методів, заснованих на багатовимірних статистиках, і обґрунтування їх ефективності для перевірки коротких послідовностей;
- створення пакету рекомендованих методів для тестування послідовностей на випадковість;
- реалізація комплексу програм для проведення тестувань бітових послідовностей.

Мета роботи – формальна та програмна реалізація тестів NIST та методів тестування, заснованих на використанні багатовимірних статистик.

Виклад основного матеріалу дослідження. Специфіка тестів описаних пакетів є такою, що на основі вхідної послідовності бітів визначається статистика, яка або є результатом, або використовується для його пошуку. Цей підхід враховує тільки одну характеристику послідовності при одному випробуванні.

У роботі запропоновано новий підхід до тестування бітової послідовності – тестування на основі багатовимірних статистик. Багатовимірні статистики орієнтовані на кілька властивостей, що дає змогу точніше оцінити коротку послідовність, але є недоліки в тестуванні довгої через надмірну кількість варіантів комбінацій статистик.

Тестування бітових послідовностей на основі багатовимірних статистик

Генератори випадкових чисел мають тенденцію до створення великої кількості повторюваних шаблонів [1]. Тести багатовимірних статистик також показують більш ефективні результати в перевірці шаблонів завдяки оцінці кількох статистик одночасно.

Методи, що представлені в роботі, засновані на дослідженні кількості входжень дво- та трибітових шаблонів у послідовність бітів. Тести на основі багатовимірних статистик у результаті виконання надають спільну вірогідність відповідної кількості шаблонів у послідовності заданої довжини. Той самий результат можна отримати за допомогою емпіричного підрахунку. Припустимо, що виконується розрахунок спільної вірогідності для всіх можливих значень $k_1 = \eta(00)$, $k_2 = \eta(111)$ та послідовності довжиною 3. Кількості входжень k_1 та k_2 до послідовності наведено в табл. 2.

Таблиця 2

Поява шаблонів у послідовності довжиною 3

Послідовність	k_1	k_2
000	2	0
001	1	0
010	0	0
011	0	0
100	1	0
101	0	0
110	0	0
111	0	1

Підрахувавши кількість появи для всіх можливих комбінацій k_1 та k_2, можна знайти відповідні вірогідності (табл. 3).

Таблиця 3

Входження шаблонів у послідовність довжиною 3

k_1	k_2	Кількість	Вірогідність
2	0	1	0,125
1	0	2	0,25
0	0	4	0,5
0	1	1	0,125

Емпіричним методом знайдено спільну вірогідність для заданої довжини і всіх можливих значень k_i, i =1, 2. Цей підхід є доволі простим і наглядно показує, для чого використовуються методи багатовимірних статистик, але не є ефективним (кількість послідовностей які необхідно перевірити при довжині $32 - 2^{32}$). Випробування, побудовані на формулах спільної вірогідності, є більш доцільними як у математичному сенсі, так і в програмному.

Тести багатовимірних статистик відрізняються тільки шаблонами, на яких перевіряється послідовність. Кожен метод отримує на вхід випадкову величину:

$$\gamma_1, \gamma_2 \ldots, \gamma_n, \qquad (1)$$

де $\gamma_i \in \{0, 1\}$, $i = 1,2\ldots,n$, $n > 0$.

Для цієї величини визначається кількість специфічних шаблонів k_1, k_2 та k_3 (якщо це визначено методом) і виконується обчислення за допомогою формули, специфічної для методу.

Розглянемо, наприклад, кілька тестів з набору «багатовимірних статистик».

Перший тест виконується, щоб знайти спільну вірогідність появи подій $k_1 = \eta(tt^*)$ та $k_2 = \eta(t1t^*) + \eta(t0t^*)$, при $t \in \{0, 1\}$, $t^* = 1 - t$:

$$P\left\{\eta\left(tt^*\right) = k_1, \eta\left(t1t^*\right) + \eta\left(t0t^*\right) = k_2\right\} = \sum_{m_1=k_1}^{n-k_1} p^{m_1} q^{m_0} \sum \prod_{i=0}^{1} C_{k_1}^{\delta_i} C_{m_i-k_1}^{k_1-\delta_i}, \qquad (2)$$

де n – довжина бітової послідовності, p – вірогідність появи t, q – вірогідність появи t^* ($q = 1 - p$), $m_0 = n - m_1$, $\sum$ – сума за всіма комбінаціями δ_0 та δ_1, так, що: $\delta_0 + \delta_1 = 2k_1 + k_2$.

Другий метод тестування знаходить спільну вірогідність появи подій $k_1 = \eta(tt^*)$ та $k_2 = \eta(ttt^*)$:

$$P\left\{\eta\left(tt^*\right) = k_1, \eta\left(ttt^*\right) = k_2\right\} = \sum_{m_1=k_1}^{n-k_1} p^{m_1} q^{m_0} C_{k_1}^{k_2} C_{m_1-k_1}^{k_2} C_{m_0}^{k_1}. \qquad (3)$$

Третій метод оцінює вірогідність появи шаблонів $k_1 = \eta(tt^*)$, $k_2 = \eta(t1t^*)$ та $k_3 = \eta(t0t^*)$:

$$P\left\{\eta\left(tt^*\right) = k_1, \eta\left(t1t^*\right) = k_2, \eta\left(t0t^*\right) = k_3\right\} = \sum_{m_1=k_1}^{n-k_1} p^{m_1} q^{m_0} C_{k_1}^{k_2} C_{k_1}^{k_3} C_{m_1-k_1}^{k_2} C_{m_0-k_1}^{k_3}, \qquad (4)$$

За допомогою четвертого методу можна визначити вірогідність подій $k_1 = \eta(tt^*)$ та $k_2 = \eta(ttt)$:

$$\text{і}\left\{\eta\left(tt^*\right) = k_1, \eta\left(ttt\right) = k_2\right\} = \sum_{m_1=k_1}^{n-k_1} p^{m_1} q^{m_0} C_{m_0}^{k_1} \times \times \sum_{i \in \{k_1, k_1+1\}} C_i^{m_1-k_2-i} - \left(m_1 - i, m_1 - i - k_2\right), \qquad (5)$$

де $-\left(a,b\right) = \begin{cases} C_{a-1}^{b-1}, \textit{якщо } a \ge b \ge 0; \\ 1, \textit{якщо } a = b = 0; \\ 0, \textit{в іншому випадку} \end{cases}$.

Проектування і розробка

Бібліотека статистичних методів

Згідно зі специфікацією NIST [1] та описаних методів багатовимірних статистик [11–13] створено бібліотеку, що надає користувачам два інтерфейси для виконання відповідних статистичних тестів мовою програмування Java. Бібліотека складається з трьох пакетів, зміст яких подано в табл. 4.

Інтерфейси для виклику методів є реалізаціями шаблону проектування «фасад» – вони групують окремі методи в один клас для уніфікації виклику методів та легкості використання. Спрощену діаграму класів для пакету NIST зображено на рис. 1. Головним інтерфейсом і «фасадом» є клас «NistTest», що одночасно виконує перенаправлення викликів до відповідних класів і перевірку вхідних параметрів за допомогою класу «Validator».

Таблиця 4

Пакети бібліотеки

Пакет	Зміст та опис пакета
nisttest	15 класів із тестами NIST, головний клас для виклику тестів
mdtest	8 класів із тестами, заснованими на багатовимірних статистиках, головний клас для виклику тестів
util	Допоміжні класи, що включають: методи перевірки вхідних параметрів тестів; допоміжні методи для роботи з шаблонами і матрицями; функції erfc, Gamma та нормального розподілу; алгоритми дискретної трансформації Фур'є і Берлекампа-Мессі та інші допоміжні функції, що використовуються в обох пакетах тестів

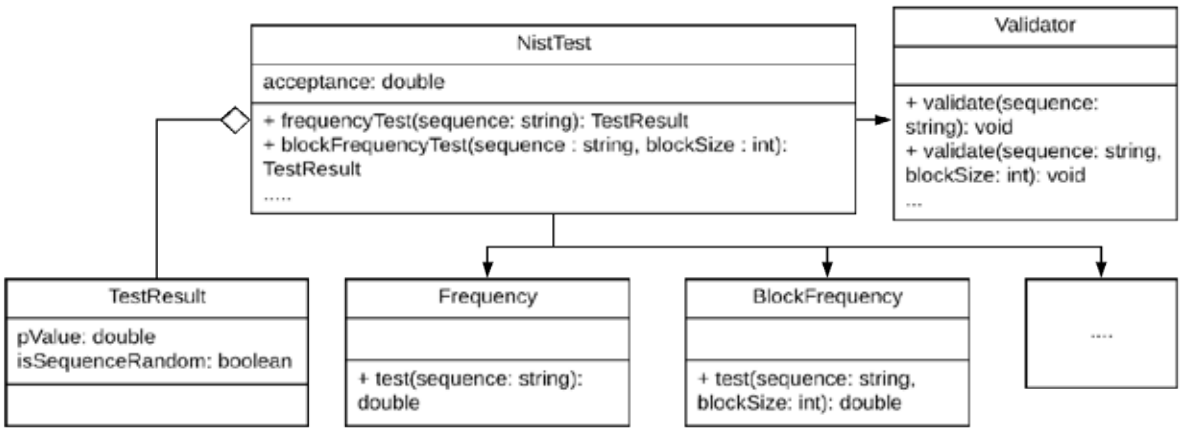

Рис. 1. Спрощена діаграма класів для тестів NIST

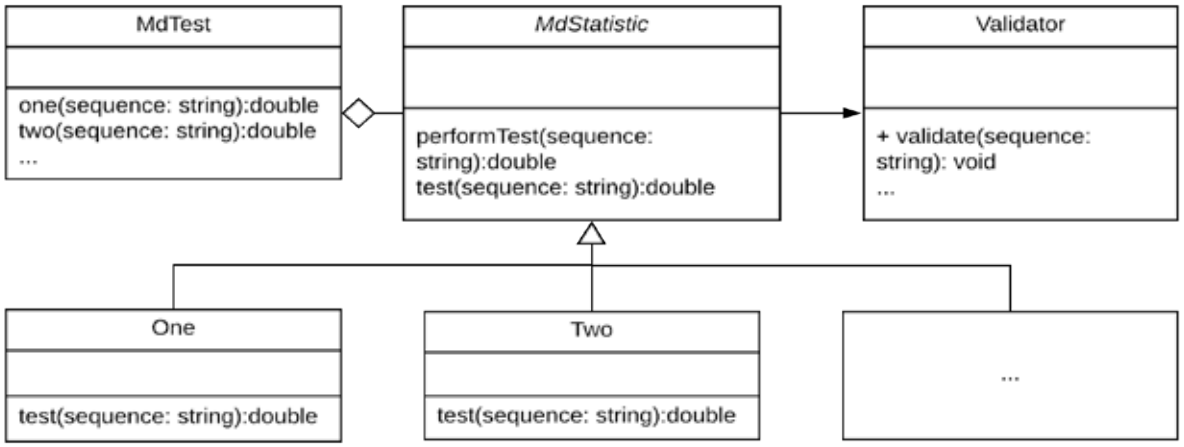

Рис. 2. Спрощена діаграма класів для тестів багатовимірних статистик

Результатом і значенням, що повертає тест, є об'єкт класу «TestResult» або список таких об'єктів, коли проводиться кілька тестів в одному методі. Кожен з окремих класів статистичних методів (наприклад, «Frequency»), використовує хоча б один клас із пакета «util» у своїй реалізації.

Окрім «фасаду», класи тестів багатовимірних статистик використовують шаблон «Шаблонний метод» для перевірки вхідних параметрів (рис. 2).

Клас «MdStatistic» містить один абстрактний метод та один звичайний, а підкласи виконують визначення абстракції. При виклику однієї функції виконання передається іншій реалізованій нащадком. Кожен із тестів використовує засоби роботи з шаблонами класу «TemplateUtils» та біноміальний розподіл або функцію Z (формула 5) з класу «FunctionUtils». Повна діаграма класів не наведена через її громіздкість.

Сервер і прикладний програмний інтерфейс

Сервер програми виконує обробку вхідних HTTP запитів (диспетчеризацію, зчитування даних із формату JSON і передачу відповідним

статистичним тестам) і створення відповідей (створення повідомлень про помилку, перетворення об'єктів на JSON, відправка відповіді). Пакетна структура сервера подана в табл. 5.

Для всіх тестів із бібліотеки було створено чотири адреси виконання статистичних тестів (табл. 6).

Під час надходження запиту виконується перевірка наявності обробника на сервері і, якщо його знайдено, запит передається відповідному контролеру, який своєю чергою викликає сервіс, що визначено в методі. Імена тестів, які необхідно виконати, визначені параметрами URL або тілом запиту. Щоб вибрати відповідний метод, у програмі створено два класи – «NistFactory» та «MdFactory». Вони реалізують шаблони проектування «Фабрика» і «Шаблонний метод» за допомогою перелічуваного типу. Архітектуру основних компонентів модуля показано на рис. 3.

Щоб забезпечити відповідний рівень параметрів надійності та живучості серверу, створено клас «ErrorHandler», який виконує перехоплення

Таблиця 5

Пакети сервера

Пакет	Зміст та опис пакета
controller	Класи «NistController» та «MdController», що визначають URL та HTTP методи, доступні на сервері і викликають відповідні методи їх обробки
service	Інтерфейси і класи, що забезпечують виконання бізнес-логіки сервера
util	Допоміжні класи, що включають конвертери перелічуваних типів даних, обробник помилкових запитів, класи для виконання методів із бібліотеки статистичних тестів

Таблиця 6

HTTP методи і адреси, доступні на сервері

Метод	Відносна адреса	Опис функціональності
POST	/nist_test/{testName}	Параметр {testName} визначає статистичний тест, який буде виконано, наприклад «/nist_test/binaryMatrix». За цими адресами виконується тільки один тест, який потребує правильних параметрів у тілі запиту
	/md_test/{testName}	
POST	/nist_test	Виконуються всі тести NIST, JSON-об'єкти з параметрами яких наявні в тілі запиту
	/md_test?tests=[testName1, testName2…]	Виконуються всі тести багатовимірних статистик, імена яких наявні в параметрах запиту для послідовності з тіла запиту

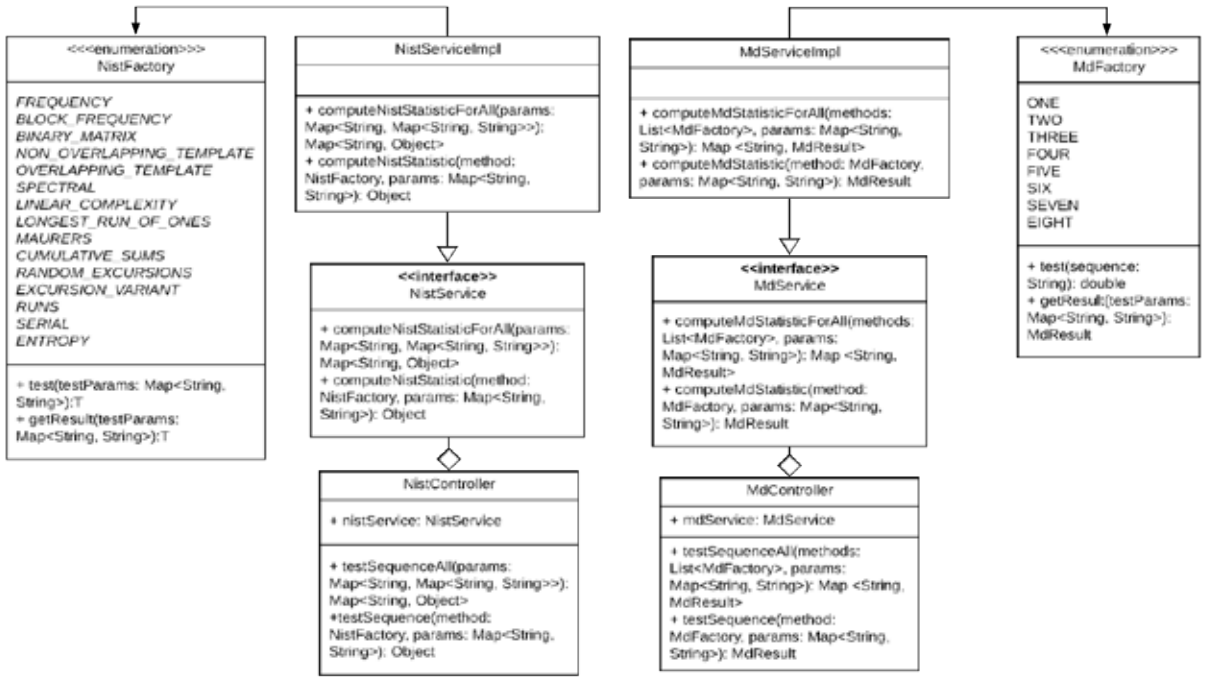

Рис. 3. Діаграма класів для основних модулів сервера

визначених виключень під час виконання запиту. Клас «ErrorEntity», що є контейнером для параметрів помилки, стандартизує вигляд і зміст повідомлень. У разі виникнення виключних ситуацій повертається HTTP код, що відповідає їх змісту, його розшифровка, текстове повідомлення, поточний час та URL. У випадку успішного виконання запиту повертається тіло з результатами тесту та код 200 (OK).

Створений сервер забезпечує прийом запитів, відправку відповідей, передачу вхідних параметрів відповідним статистичним тестам і обробку всіх можливих помилкових ситуацій.

Опис засобу

Метою вебдодатку є забезпечити користувачів можливістю тестування бітових послідовностей на випадковість за допомогою графічного інтерфейсу. Він має забезпечити високий рівень зручності використання: надавати свободу дій, врахувати можливість помилок, повідомляти інформацію про стан системи та містити довідкові матеріали.

Основний функціонал, який має забезпечувати додаток, подано на діаграмі варіантів використання (рис. 4). Відповідно до діаграми, між задачами, що виконує вебдодаток, та задачами сервера є чіткий розподіл. Перший працює незалежно більшу частину часу і викликає другого, тільки коли не може виконати поставлену задачу самостійно.

Компоненти розроблюваного засобу можна умовно поділити на кілька груп:

− компоненти тестів NIST визначають елементи форми введення даних, адресу для прикладного інтерфейсу, довідковий матеріал до тестів і засоби перевірки вхідних параметрів;

− компоненти багатовимірних статистик визначають адресу для отримання даних із сервера та довідковий матеріал;

− компоненти форм групують спільні характеристики методів для того, щоб прибрати повторюваність коду. Забезпечують перевірку і форматування параметрів за визначеними методами, реалізують створення, відправку та обробку відповіді HTTP запиту;

− компоненти меню визначають адреси, за якими знаходяться всі сторінки, забезпечують відображення посилань на сторінках і правильний перехід за ними;

− компоненти графіків забезпечують побудову графіків відповідно до результатів тестування послідовності;

− допоміжні компоненти підтримують уніфіковану структуру результатів тестування та логіку переходу за посиланнями;

− валідатор об’єднує логіку перевірки вхідних параметрів, щоб знизити повторюваність коду;

− стилі CSS визначають зовнішній вигляд сторінок.

Вебдодаток

У процесі переходу на сайт із пошукової системи користувача буде направлено на головну сторінку (рис. 5).

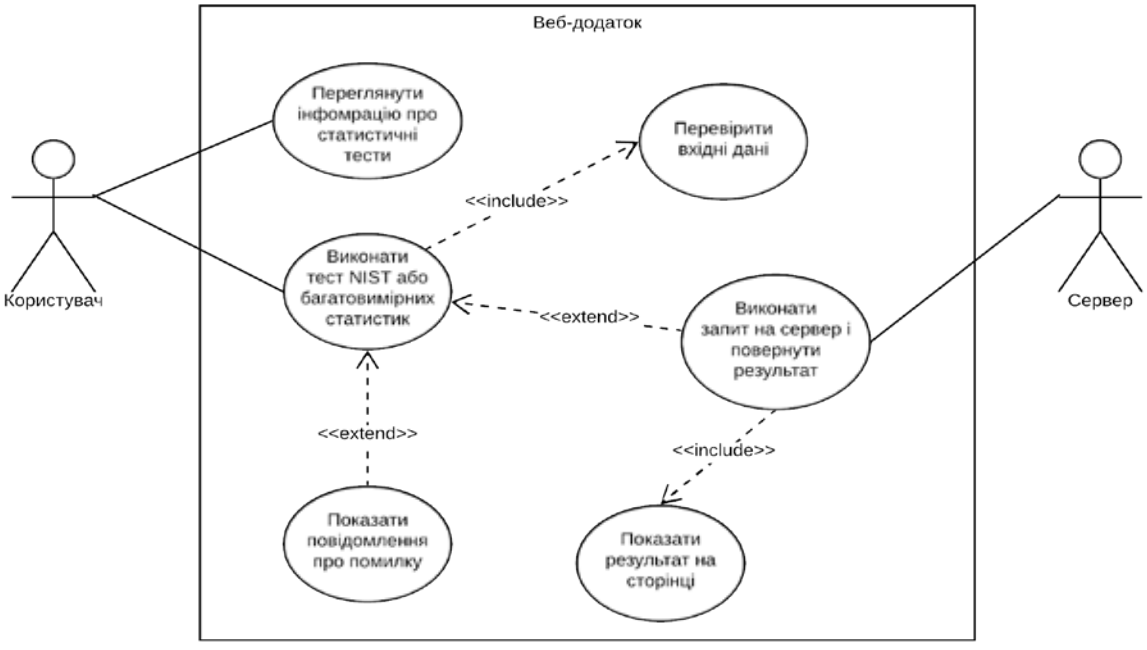

Рис. 4. Діаграма варіантів використання

Рис. 5. Головна сторінка

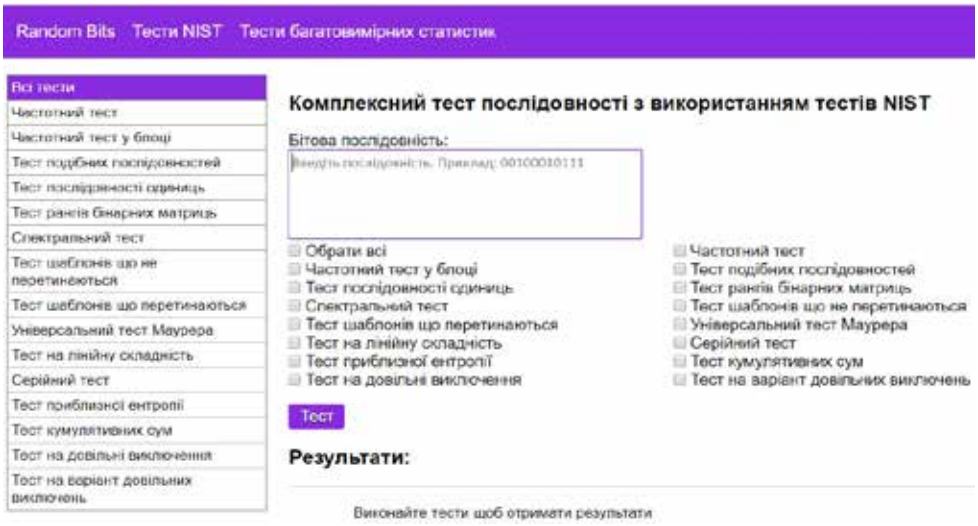

Рис. 6. Сторінка NIST

Головна сторінка надає довідкову інформацію про зміст вебдодатку та корисні посилання. Використавши верхню навігаційну панель, можна перейти на сторінки, що відповідають тестам NIST (рис. 6) та багатовимірним статистикам.

За замовчуванням відкриваються комплексні тести, тобто для одночасного виконання кількох методів.

Кожна форма містить елементи управління «Checkbox» для вибору окремих тестів та вибору/зняття всіх одночасно. У текстові поля вводиться послідовність бітів.

Бокові панелі на обох сторінках містять меню, в якому можна вибрати один із тестів.

Висновки. Тестування бітової послідовності не є новою проблемою. Нині є велика кількість пакетів тестів, що вирішують цю задачу. Однак специфіка екосистеми тестування та проблеми наявних методів вказують на необхідність покращення підходів.

Тести багатовимірних статистик дають змогу краще дослідити послідовність шляхом використання одночасно кількох характеристик послідовності. Вони засновані на дослідженні шаблонів довжин 2 та 3 і допомагають виявляти приховані залежності між даними та неякісні генератори. Головною перевагою тестів є їх ефективність на послідовностях короткої довжини.

Пакет програм надає кілька можливих рівнів використання залежно від вимог користувача і складається з:

− бібліотеки мовою Java, що включає 15 тестів NIST та 8 тестів багатовимірних статистик;

− прикладний програмний інтерфейс, що дає змогу використовувати тести за допомогою HTTP запитів;

− вебдодаток, який може бути використано для тестування послідовностей через браузер.

Пакет тестів рекомендовано до використання в процесі дослідження послідовностей на випадковість. Вони можуть бути застосовані в одній з областей:

− наукові дослідження – встановлення залежності між будь-якими експериментальними даними, розробка ГПЧ та АГВЧ, створення нових методів перевірки послідовності на випадковість;

− криптографія – перевірка послідовностей згенерованих ГПЧ та АГВЧ, дослідження алгоритмів шифрування;

− розроблення та супровід програмних продуктів – тестування ефективності алгоритмів та систем, заснованих на випадковості, перевірка криптографічних засобів системи.

Список літератури:

1. Rukhin A. A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications. *National Institute of Standards and Technology.* 2010. URL: https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecial publication800-22r1a.pdf.
2. DIEHARD Statistical Tests. URL: https://stat.fsu.edu/pub/diehard/.
3. Diehard tests. URL: https://en.wikipedia.org/wiki/Diehard_tests.
4. TestU01: A software library in ANSI C for empirical testing of random number generators. *Department d'Informatique et de Recherche Operationnelle, University of Montreal*. 2013. URL: http://simul.iro.umontreal.ca/testu01/guideshor ttestu01.pdf.
5. Иванов М.А., Чугунков И.В. Теория, применение и оценка качества генераторов псевдослучайных последовательностей. Москва : Кудиц-образ, 2003. 240 с.
6. Акимова Г.П., Пашкина Е.В., Соловьев А.В. Методологический подход к оценке качества случайных чисел и последовательностей. *Труды ИСА РАН.* 2008. Т. 38. С. 156–167.
7. Кнут Д. Искусство программирования. Т. 2. Получисленные алгоритмы. Москва, 2000. 832 с.
8. Лагутин М.Б. Наглядная математическая статистика : учеб. пособие. 5-е изд. (эл.). Москва : БИНОМ. Лаборатория знаний, 2015. 475 с. URL: https://rucont.ru/efd/321098
9. The RAND Corporation. A Million Random Digits with 100 000 Normal Deviates. Free Press, N.Y. 1966.
10. Гайдышев И.П. Программное обеспечение анализа данных AtteStat. Руководство пользователя. Версия 13. 2012. 505 с.
11. Svitlana Popereshnyak, Georgi P. Dimitrov The Testing of Pseudorandom Sequences using Multidimensional Statistics Proceedings of the 1st International Workshop on Digital Content & Smart Multimedia (DCSMart 2019) Lviv, Ukraine, December 23–25, 2019. P. 151–161
12. Masol V.I., Popereshnyak S.V. Statistical analysis of local sections of bits sequences. *Journal of Automation and Information Sciences.* 2019. Vol. 51. P. 31–45. DOI: 10.1615/JAutomatInf Scien.v51.i10.30
13. Masol V.I., Popereshnyak S.V. Checking the Randomness of Bits Disposition in Local Segments of the (0, 1)-Sequence. *Cybernetics and Systems Analysis.* 2020. 56(3). P. 1–8. DOI: 10.1007/s10559-020-00267-0

Popereshnyak S.V. SOFTWARE FOR TESTING BIT SEQUENCES FOR RANDOMNESS

The article systematizes scientific positions on static testing of sequences, which are widely used in cryptographic information security systems for the production of key and additional information. A brief overview of known methods of random sequence testing, as well as static tests developed in recent decades. The perspective direction of research is considered – static testing of sequences with use of multidimensional statistics is considered. The paper presents formulas for testing random bit sequences on randomness, using two-dimensional or three-dimensional statistics that can be used to test short and medium sequences.

The approach to testing using multidimensional statistics allows you to rely on a deeper justification of the randomness of the generated sequences. The advantage of testing with multidimensional statistics is also the possibility of a high degree of parallelism in the calculation of the family of statistics. This area is promising for research, especially for short and medium-length sequences, where one-dimensional statistics are often powerless.

The method of pseudo-random sequence testing is given in the paper and an explicit form of joint distribution of numbers of 2-chains and numbers of 3-chains of different variants in a random bit sequence of a given small length is obtained.

To implement the proposed technique, a software tool was developed to test the bit sequence for randomness. This tool includes NIST tests, as well as tests using multidimensional statistics, which have proven themselves well when testing a pseudo-random sequence of short length. As a result of further development, a single information system will be expanded and created, which will allow to analyze the bit sequence using known tests and / or tests using multidimensional statistics and choose a high-quality pseudo-random sequence for use in a particular subject area.

Key words: *software, bit sequence, testing, multidimensional statistics; random sequences; pseudo-random sequence; statistical testing.*