УДК 004.056.5 **DOI:10.30837/rt.2022.2.209.02**

О.О. КУЗНЕЦОВ, д-р техн. наук, Ю.І. ГОРБЕНКО, канд. техн. наук,
М.О ПОЛУЯНЕНКО, канд. техн. наук, С.О. КАНДІЙ, Є.Д. МАТВЄЄВА

ВЛАСТИВОСТІ ФУНКЦІЇ ВАРТОСТІ В ІТЕРАТИВНОМУ АЛГОРИТМІ ГЕНЕРАЦІЇ НЕЛІНІЙНИХ ПІДСТАНОВОК

Вступ

При проєктуванні шифру із симетричним ключем складну задачу становить генерація криптографічно стійких нелінійних підстановок (S-блоків) [1 – 3]. По перше, нелінійні підстановки повинні бути випадковими, тобто не містити простих алгебраїчних конструкцій, бо це може створити передумови для ефективного алгебраїчного криптоаналізу [4, 5]. По друге, S-блоки повинні забезпечувати необхідні криптографічні властивості, які значно ускладнюють реалізацію різних криптоаналітичних атак (диференційного, лінійного, статистичного та інш.) [3, 6, 7]. Отже задача генерації нелінійних підстановок є складною та надзвичайно важливою з точки зору подальшого удосконалення криптографічних алгоритмів із симетричним ключем.

Найбільш перспективними для генерації високонелінійних S-блоків вважаються евристичні техніки. Вони дозволяють ітеративним шляхом змінювати початковий випадковий S-блок доти він не буде відповідати встановленим критеріям. Однак час такої генерації може бути занадто великим. Наприклад, для кращого відомого результату генерація випадкових бієктивних 8-бітних підстановок із нелінійністю понад 104 вимагає понад 65 тисяч ітерацій [8, 9]. Метою цієї роботи є оптимізація евристичних методів для прискорення генерації високонелінійних S-блоків.

Пов’язані роботи

В цій роботі розглядаються евристичні техніки генерації нелінійних підстановок. До таких алгоритмів відносяться евристичні методи:

- алгоритм локального пошуку (Local Search Algorithm) [1, 8 – 10];
- алгоритм сходження на пагорб (Hill climbing) [8, 11 – 13];
- метод градієнтного спуску (gradient descent method) [6, 14];
- алгоритм імітації відпалу (simulated annealing) [12, 15 – 17] та [10, 18];
- генетичний алгоритм (Genetic Algorithm) [19 – 21] тощо.

Основне завдання евристичних технік – зниження (або у деяких випадках збільшення) функції вартості, яка пов’язана з бажаною властивістю S-блоку. Під час роботи алгоритму пошуку виконується наближення характеристик поточного S-блоку до бажаного значення. Слід зазначити, що успіх генерації дуже чутливий до обраної функції вартості, а отже до вибору її параметрів. Серед відомих функції вартості слід виділити такі найбільш популярні:

- функція вартості Кларка (Clark’s cost functions *WHS*) [15] та її модифікація [10, 18];
- функція вартості Пічека (Picek’s cost functions *PCF*) [8, 22];
- функція вартості Фрейре – Ечеваррія (Freyre – Echevarría cost functions *WCF*) [8, 9].

У даній роботі розглядається функція вартості WCF, яка була запропонована у роботах [8,9]. Використовуючи алгоритм сходження на пагорб (Hill climbing) [2, 11, 23] та функцію WCF авторами було отримано кращий відомий на сьогоднішній день результат з генерації 8-бітних бієктивних підстановок із нелінійністю 104 [8, 9]. Середня кількість ітерацій алгоритму пошуку до знаходження S-блоку з нелінійністю 104 становила 65,933 [9]. В [9] наведено, що з 30 незалежних експериментів у 11 випадках було знайдено S-блок з нелінійністю 104. У іншій публікації тих самих авторів [8] наведено середнє значення у 70,596 ітерацій.

В цій роботі перевіряються результати робіт [8, 9] та оптимізуються параметри функції WCF. Зокрема, ми підтверджуємо результати [8, 9] та показуємо, що функція WCF може

бути ще ефективнішою. В наших експериментах ми отримали найменше значення числа ітерацій для функції WCF та Hill climbing алгоритму. Фактично нам вдалося значно підвищити ефективність евристичного пошуку через зменшення кількості ітерацій.

Методика дослідження

Для пошуку бієктивних S-блоків з високою нелінійністю в цій роботі використовується алгоритм Hill climbing [8, 11 – 13]. Сходження на пагорб – це ітераційний алгоритм, який починає свій пошук з деякої можливої точки, випадково обраної в просторі станів. Потім послідовно застосовується механізм генерації для пошуку кращого рішення (з точки зору значення функції вартості), досліджуючи сусідство поточного рішення. Якщо знайдено краще рішення, воно стає поточним рішенням. Алгоритм закінчує роботу, коли не вдається знайти покращення, а поточне рішення розглядається як приблизне рішення задачі оптимізації.

Алгоритм сходження на пагорб оптимізує функцію вартості, досліджуючи сусідні точки рішення відносно поточної точки в просторі рішень. Нижче розглядаємо (S, f) приклад комбінаторної задачі оптимізації (де S – набір можливих рішень; f – функція вартості, яку слід мінімізувати).

Алгоритм сходження на пагорб (для задачі мінімізації) можна узагальнити наступним псевдокодом 1:

Псевдокод 1. Пошук локального мінімуму

1. Вибрати початкове рішення S_i;
2. Генерувати рішення S_j із сусідства поточного рішення S_i;
3. Якщо ($f(S_j) < f(S_i)$), то S_j стає поточним рішенням;
4. Якщо ($f(S_j) \geq f(S_i)$) для певної кількості S_j, то закінчити;
5. Перейди до кроку 2.

В роботі було запрограмовано алгоритм сходження на пагорб, який одночасно виконував пошук в декількох потоках, працюючих паралельно. Кількість потоків вказується у вхідному параметрі thread_count (в нашому випадку thread_count = 2). Алгоритм починає свою роботу з підстановки, яку згенеровано випадково. Ця підстановка встановлюється як поточне рішення S_i. Поточне рішення є загальним для всіх потоків. На кожній ітерації циклу утворюється декілька (відповідно до параметру thread_count) нових рішень S_j, які генеруються за заданими операторами мутації. Оператор мутації обирає випадковим чином $k = 2$ різних позицій у підстановці S_i і переставляє елементи у обраних позиціях. Нове рішення S_j порівнюється з поточним S_i. В разі отримання кращого, ніж поточне, рішення воно встановлюється як поточне.

Всі ітерації пошуку за алгоритмом сходження на пагорб виконуються у внутрішньому циклі. Ітерації внутрішнього циклу вкладено в зовнішній цикл. Зовнішній цикл не є обов'язковим для роботи алгоритму, він введений лише для відстеження поточного стану процесу пошуку та оптимізації вибору його параметрів. Докладніше алгоритм розглянуто у [10].

В якості цільового S-блоку було обрано бієктивний 8-бітний S-блок з нелінійністю $N_f = 104$. В якості інших параметрів використовувались наступні:

- кількість внутрішніх циклів – max_inner_loops=10000;
- максимальна кількість зовнішніх циклів – max_outer_loops=50;

- максимальна кількісті поспіль зовнішніх циклів, при яких не виконано жодного покращення функції вартості – max_frozen_outer_loops=5.

Критерії зупинки алгоритму:

- знаходження бієктивного S-блоку з нелінійністю 104;
- досягнення максимальної кількості ітерацій (відповідає значенню thread_count x max_inner_loops x max_outer_loops);
- досягнення кількості поспіль зовнішніх циклів, при яких не виконано жодного покращення функції вартості значення max_frozen_outer_loops.

Дослідження функції вартості *WCF*

Як основний апарат аналізу та вивчення особливостей критеріїв зручно вибрати перетворення Фур'є та Уолша булевих функцій [1, 11, 24].

Перетворення Уолша – Адамара

Позначимо через **x, ω** двійкові набори довжини n над $GF(2)$, а x_i, ω_i – координати цих наборів. Якщо, $f(x_1, x_2, \ldots, x_n)$ – булева функція двійкових змінних, то через $f'(x_1, x_2, \ldots, x_n) = (-1)^{f(x_1, x_2, \ldots, x_n)}$ позначимо спряжену функцію, визначену на тій самій множині. Функції f та f' однозначно визначають один одного. Скалярний добуток **x** та **ω** – це цілочисельна функція, яка визначається як

$$\langle \mathbf{x}, \boldsymbol{\omega} \rangle = \sum_{i=1}^{n} x_i \cdot \omega_i \ .$$

Перетворення Уолша булевої функції $f(\mathbf{x})$ позначається, як

$$W_f\left(f(\mathbf{x}), \omega\right) = \sum_{\mathbf{x}=F_2^n} f(\mathbf{x}) \cdot (-1)^{\langle \mathbf{x}, \omega \rangle} \ .$$

Спектральне перетворення функції $f'(\mathbf{x})$ позначається через

$$WHT\left(f(\mathbf{x}), \omega\right) = \sum_{\mathbf{x}=F_2^n} (-1)^{f(\mathbf{x}) + \langle \mathbf{x}, \omega \rangle},$$

та носить назву перетворення Уолша – Адамара булевої функції.

Спектральні перетворення дозволяють безпосередньо оцінити збалансованість, нелінійність та кореляційну імунність булевої функції [1, 11, 24]. Зокрема, нелінійність S-блоку виражається як

$$N(S) = 2^{n-1} - \frac{1}{2} \cdot \max\left(|WHT|\right), \tag{1}$$

де $\max\left(|WHT|\right)$ – максимальне абсолютне значення в спектрі Уолша-Адамара за всіма компонентними булевими функціями S-блоку.

Розподіл спектральних коефіцієнтів Уолша – Адамара

Приклад значень спектральних коефіцієнтів Уолша – Адамара для випадково сформованого бієктивного 8x8 S-блоку, представлено на рис. 1 (представлено фрагмент для 256 значень).

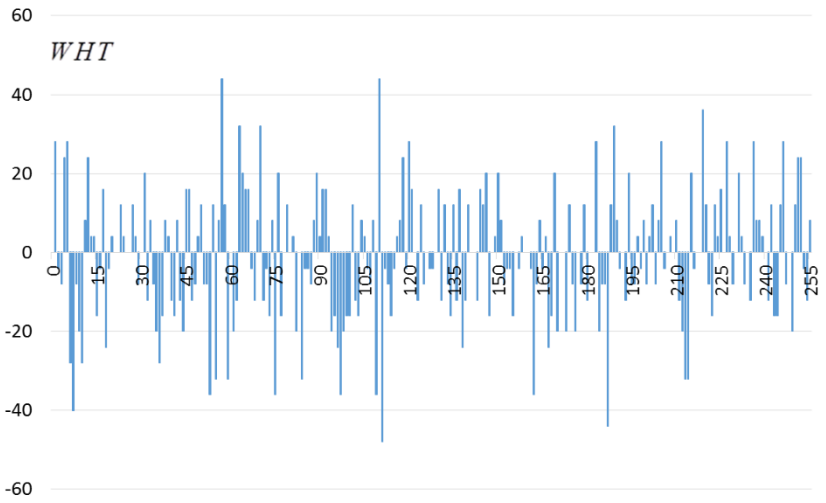


Рис. 1. Перші 256 спектральних коефіцієнтів Уолша – Адамара для випадково сформованого бієктивного S-блоку (приклад)

Як бачимо для даного прикладу, значення WHT змінюються від – 48 до +57. Зміна значень WHT завжди відбувається з шагом 4. Гістограма розподілу кількості коефіцієнтів WHT за їх значеннями (для всіх 65 280 значень) наведена на рис. 2. По лінії абсцис відкладено значення, яке приймає WHT, а за ординат – кількість випадків, коли у спектрі з'являється таке значення WHT.

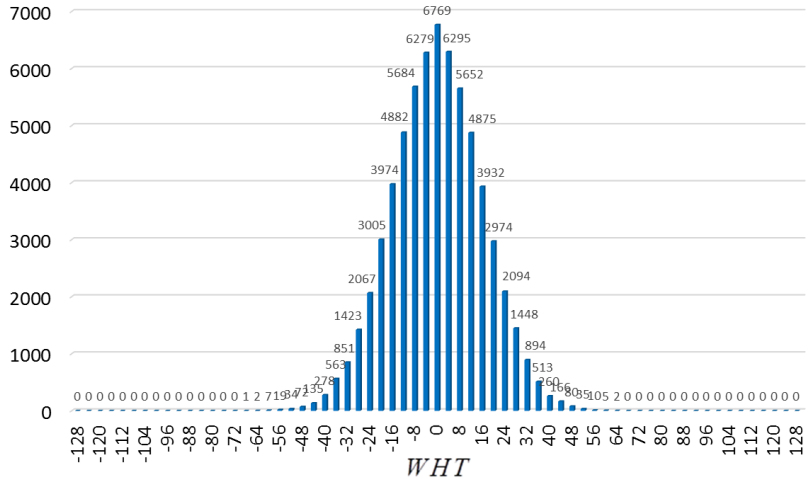


Рис. 2. Розподіл значень спектральних коефіцієнтів Уолша – Адамара для випадково сформованого бієктивного S-блоку (приклад)

З урахуванням (1) нас цікавить максимальне значення спектру, тобто $\max\left(\left|WHT\right|\right)$. У наведеному прикладі це буде -68, тобто $N(S)=94$.

При використанні евристичних алгоритмів пошуку цільового S-блоку здійснюється поступове зменшення $\max\left(\left|WHT\right|\right)$, що призводить до підвищення нелінійності S-блоку. Так, на рис. 3 наведено фінальний розподіл кількості WHT за їх значеннями при $N(S)=104$. На рис. 4 наведено гістограму змін розподілу кількості WHT від начального стану (випадково сформованого бієктивного S-блоку) до кінцевого стану. В цьому експерименті застосовувався алгоритм сходження на пагорб (Hill climbing). Всього було виконано 117 покращень функції *WCF*. Символом k позначено кількість прийнятих покращень у алгоритмі пошуку.

Як бачимо з наведених результатів, форма розподілу та його максимум суттєво не змінюється під час покращень за обраним алгоритмом пошуку. Тому, цілком доречно виглядає ідея враховувати лише частину розподілу спектру коефіцієнтів Уолша – Адамара, яка наближена до $\max\left(\left|WHT\right|\right)$, що і було реалізовано у функції вартості *WCF*.

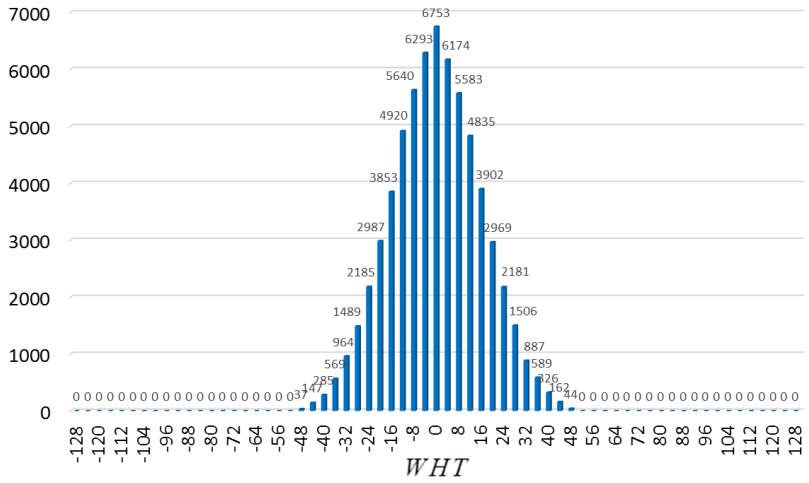


Рис. 3. Розподіл значень спектральних коефіцієнтів Уолша – Адамара для отриманого бієктивного S-блоку з нелінійністю $N(S)=104$

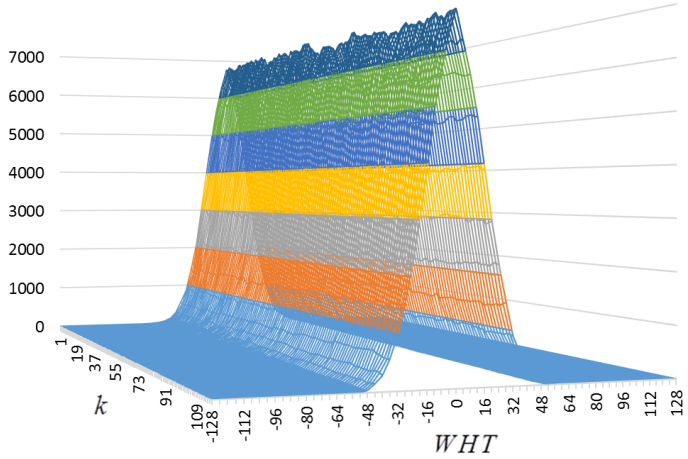


Рис. 4. Гістограма змін розподілу значень спектральних коефіцієнтів Уолша – Адамара

Опис функції вартості WCF

Функція вартості *WCF* у загальному випадку має наступний вигляд [8, 9]:

$$WCF=\sum_{b=1}^{255}\sum_{i=0}^{255}\prod_{\substack{j=start\\ j+=step}}^{end}\left|\left|WHT[b,i]\right|-j\right|, \tag{2}$$

де

- *WHT* – спектральні коефіцієнти Уолша – Адамара;
- $start, step, end$ – деякі цілі значення, як правило $start=0, step=4$ (виходячи з кратності коефіцієнтів *WHT* чотирма);
- i – змінна циклу за всіма компонентними функціями та їх лінійними комбінаціями;
- b – змінна циклу за всіма лінійними функціями.

Для кожного S-блоку розміру 8x8 є $256 \cdot 256 = 65536$ значень спектральних коефіцієнтів Уолша – Адамара. Причому, при $b = 0$ перше значення завжди буде дорівнювати 256, а наступні 255 – нулю, тому сума починається з одиниці та загальна кількість коефіцієнтів, що досліджується, становить 65 280.

Гістограма зміни значення функції *WCF*, яка відповідає зміни розподілу спектральних коефіцієнтів Уолша – Адамара, наведена на рис. 5. Цю діаграму отримано при параметрах $start = 0, step = 4, end = 32$.

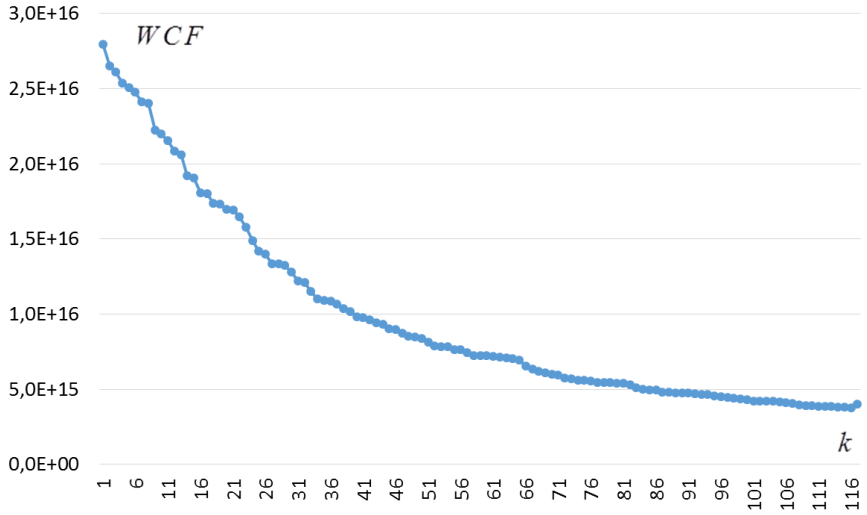


Рис. 5. Діаграма зміни значення функції *WCF*

Функція *WCF* фактично приймає за нульовий вклад значень спектральних коефіцієнтів Уолша – Адамара та враховує лише їх крайні значення, індекси яких за модулем більші ніж значення *end*. Наочно це представлено на рис. 6, який отримано зі значень розподілу, наведених на рис. 3. Тут застосовано обмеження, які використовуються в підрахунку функції *WCF* та представлено у логарифмічному масштабі.

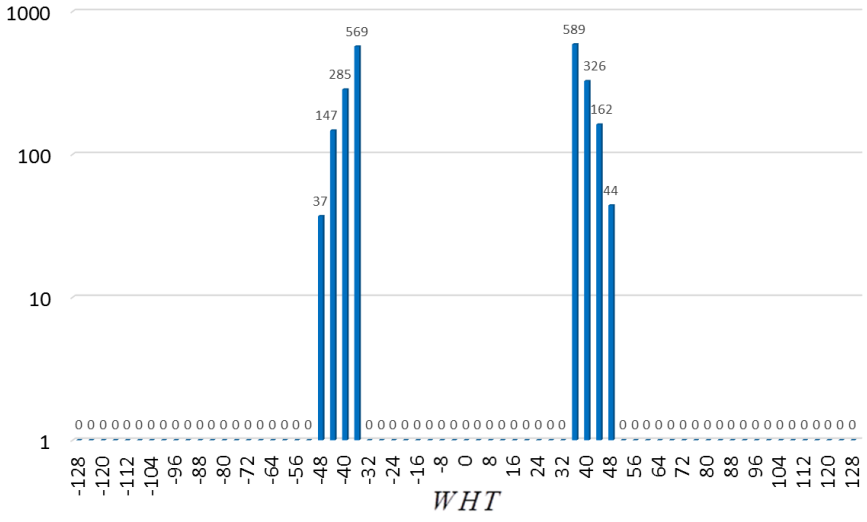


Рис. 6. Розподіл значень спектральних коефіцієнтів Уолша – Адамара які враховуються у функції *WCF* для отриманого бієктивного S-блоку з нелінійністю $N_f = 104$ (приклад)

Розрахунок функції (2) для наведеного прикладу буде проводитися наступним чином:

$$WCF = \sum_{b=1}^{255}\sum_{i=0}^{255}\prod_{\substack{j=start\\ j+=step}}^{end} \| WHT[b,i] | - j | =$$

$$= 37\cdot\left[\left(|-48|-0\right)\cdot\left(|-48|-4\right)\cdot\left(|-48|-8\right)\cdot\left(|-48|-12\right)\cdot\left(|-48|-16\right)\cdot\right.$$
$$\left.\cdot\left(|-48|-20\right)\cdot\left(|-48|-24\right)\cdot\left(|-48|-28\right)\cdot\left(|-48|-32\right)\right]+$$
$$+147\cdot\left[\left(|-44|-0\right)\cdot\left(|-44|-4\right)\cdot\left(|-44|-8\right)\cdot\left(|-44|-12\right)\cdot\left(|-44|-16\right)\cdot\right.$$
$$\left.\cdot\left(|-44|-20\right)\cdot\left(|-44|-24\right)\cdot\left(|-44|-28\right)\cdot\left(|-44|-32\right)\right]+$$
$$+285\cdot\left[\left(|-40|-0\right)\cdot\left(|-40|-4\right)\cdot\left(|-40|-8\right)\cdot\left(|-40|-12\right)\cdot\left(|-40|-16\right)\cdot\right.$$
$$\left.\cdot\left(|-40|-20\right)\cdot\left(|-40|-24\right)\cdot\left(|-40|-28\right)\cdot\left(|-40|-32\right)\right]+$$
$$+569\cdot\left[\left(|-36|-0\right)\cdot\left(|-36|-4\right)\cdot\left(|-36|-8\right)\cdot\left(|-36|-12\right)\cdot\left(|-36|-16\right)\cdot\right.$$
$$\left.\cdot\left(|-36|-20\right)\cdot\left(|-36|-24\right)\cdot\left(|-36|-28\right)\cdot\left(|-36|-32\right)\right]+$$
$$+589\cdot\left[\left(|36|-0\right)\cdot\left(|36|-4\right)\cdot\left(|36|-8\right)\cdot\left(|36|-12\right)\cdot\left(|36|-16\right)\cdot\right.$$
$$\left.\cdot\left(|36|-20\right)\cdot\left(|36|-24\right)\cdot\left(|36|-28\right)\cdot\left(|36|-32\right)\right]+$$
$$+326\cdot\left[\left(|40|-0\right)\cdot\left(|40|-4\right)\cdot\left(|40|-8\right)\cdot\left(|40|-12\right)\cdot\left(|40|-16\right)\cdot\right.$$
$$\left.\cdot\left(|40|-20\right)\cdot\left(|40|-24\right)\cdot\left(|40|-28\right)\cdot\left(|40|-32\right)\right]+$$
$$+162\cdot\left[\left(|44|-0\right)\cdot\left(|44|-4\right)\cdot\left(|44|-8\right)\cdot\left(|44|-12\right)\cdot\left(|44|-16\right)\cdot\right.$$
$$\left.\cdot\left(|44|-20\right)\cdot\left(|44|-24\right)\cdot\left(|44|-28\right)\cdot\left(|44|-32\right)\right]+$$
$$+44\cdot\left[\left(|48|-0\right)\cdot\left(|48|-4\right)\cdot\left(|48|-8\right)\cdot\left(|48|-12\right)\cdot\left(|48|-16\right)\cdot\right.$$
$$\left.\cdot\left(|48|-20\right)\cdot\left(|48|-24\right)\cdot\left(|48|-28\right)\cdot\left(|48|-32\right)\right] = 4\ 003\ 221\ 743\ 861\ 760.$$

Отримали значення $WCF = 4\ 003\ 221\ 743\ 861\ 760$, що відповідає останньому значенні (при $k = 117$) на рис. 5.

З наведеного прикладу розрахунків бачимо:

1) Спектральні коефіцієнти Уолша – Адамара з більш великими абсолютними значеннями мають значно більшу вагу ніж їх «сусідній» менший коефіцієнт. Наприклад, збільшення на одне значенням кількості спектральних коефіцієнтів зі значенням 48 урівноважується зменшенням кількості спектральних коефіцієнтів зі значенням 44 на 4 одиниць, або зі значенням 40 на 22 одиниці, або зі значенням 36 на 220 одиниць;

2) На відміну від функції вартості *WHS*, яку було запропоновано Кларком у роботі [15], функція *WCF* не враховує центральні значення у розподілі спектру. Тобто, навить дуже значні зміни в розподілі WHT, значення яких менш ніж end, не будуть враховані у функції *WCF*;

3) Має значення не лише максимальне значення спектру коефіцієнтів Уолша – Адамара, а й наступні за максимальними значеннями. Так, з двох S-блоків, які мають однакові максимальні значення спектру коефіцієнтів Уолша – Адамара, для наступного пошуку буде обраний той S-блок, який має менші значення інших значень спектру коефіцієнтів;

4) Значення функції *WCF* дуже швидко зростає. Наприклад, при обранні $end = 28$ значення функції *WCF* випадково сформованого S-блоку складає близько $1\cdot10^{15}$, при $end = 32$ близько $2\cdot10^{16}$, при $end = 36$ близько $5\cdot10^{17}$, а вже при $end = 40$ значення функції WCF

перевищує 64-бітне значення (який охоплює діапазон від -9 223 372 036 854 775 808 до 9 223 372 036 854 775 807) та потребує застосування більш довгої арифметики, що потенційно зменшує продуктивність роботи алгоритму пошуку.

Результати дослідження та оптимізації

Модифікація функції WCF

Для зменшення швидкості зростання функції *WCF* та можливість її застосування для $end \geq 40$ з 64-бітними цілими числами, враховуючи кратність спектру коефіцієнтів Уолша – Адамара чотирьом, можна зменшити кожний множник у чотири рази. Програмно це можна виконати побітним зсувом чисел без якісної зміни поводження функції *WCF*. Тобто у подальшому будемо досліджувати модифіковану функцію вартості виду

$$WCF = \sum_{b=1}^{255} \sum_{i=0}^{255} \prod_{\substack{j=start \\ j+=step}}^{end} \frac{1}{4} \cdot \left| \left| WHT[b,i] \right| - j \right|. \qquad (3)$$

Наведена модифікація зменшує значення *WCF* у $4^{(end-start)/step}$ рази. При цьому значення функції (4) для випадково сформованого S-блоку буде складати:

- для параметру $end = 28$ близько $1 \cdot 10^{10}$,
- для параметру $end = 32$ близько $1 \cdot 10^{11}$,
- для параметру $end = 36$ близько $5 \cdot 10^{11}$,
- для параметру $end = 40$ близько $5 \cdot 10^{12}$,
- для параметру $end = 44$ близько $1 \cdot 10^{13}$,
- для параметру $end = 48$ близько $5 \cdot 10^{13}$,
- для параметру $end = 52$ близько $1 \cdot 10^{14}$,
- для параметру $end = 56$ близько $5 \cdot 10^{14}$.

Задавати значення $end > 48$ практичного сенсу немає, так як це буде відповідати $N(S) < 128 - 48/2 = 104$, тобто для $N(S) = 104$ значення функції вартості *WCF*=0.

Дослідження впливу параметру *end*

Для проведення досліджень впливу параметрів на значення функції вартості (3) здійснено тестування. При цьому параметр *end* змінювали у діапазоні від 0 до 48 з шагом 4. Параметри $start = 0$ та $step = 4$ змінювати не має сенсу через значення, які може приймати спектр коефіцієнтів Уолша – Адамара.

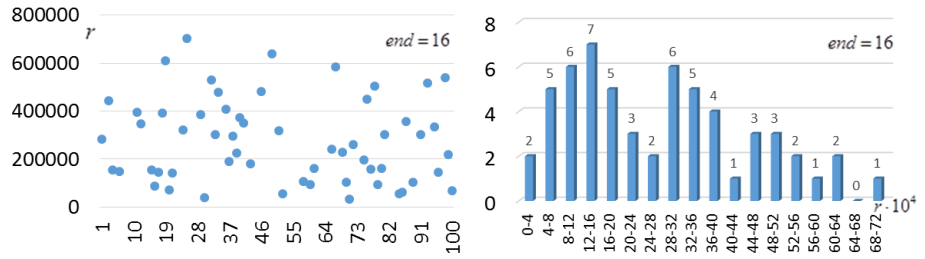
Всього було проведено 100 іспитів (окремих запусків програми для знаходження цільового S-блоку) для кожного значення *end*. Для кожної серії іспитів розраховували:

- кількість вдалих іспитів, тобто випадків знаходження цільового S-блоку (бієктивної 8-бітної підстановки з нелінійністю $N(S) = 104$);
- кількість ітерацій до знаходження цільового S-блоку (що є пропорційним до часу, затраченого на пошук);
- послідовність змін прийнятих значень функції *WCF* у алгоритмі пошуку та поточне значення $N(S)$.

На рис. 7 – 14 наведено результати тестування:

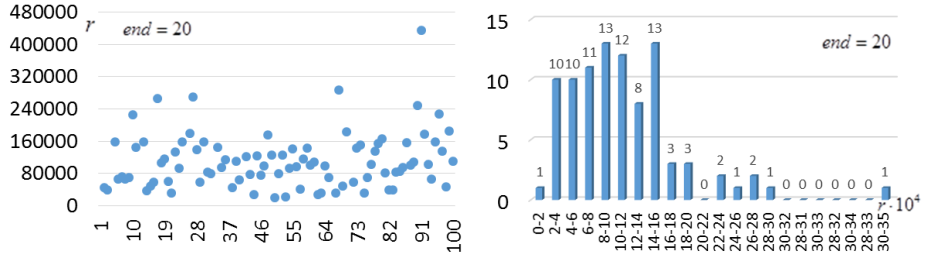
- *а* – кількість ітерацій r, які біло виконано в кожному іспиті, до знаходження цільового S-блоку;
- *б* – розподіл кількості ітерацій r.

Позначимо середнє значення кількість ітерацій символом r^{aver}. Узагальнені результати тестування наведено у табл. 1.



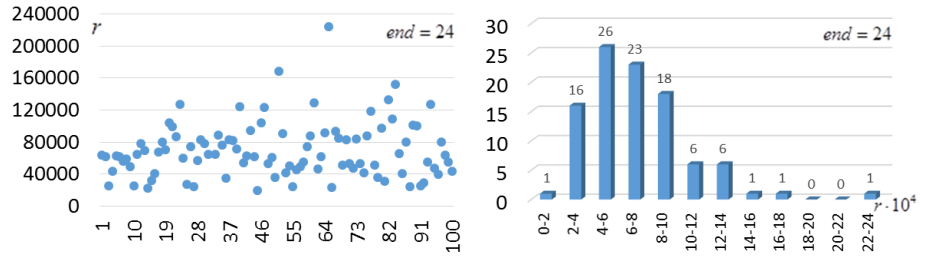
а *б*

Рис. 7. Результати тестування алгоритму Hill climbing з функцією вартості WCF, $end = 16$



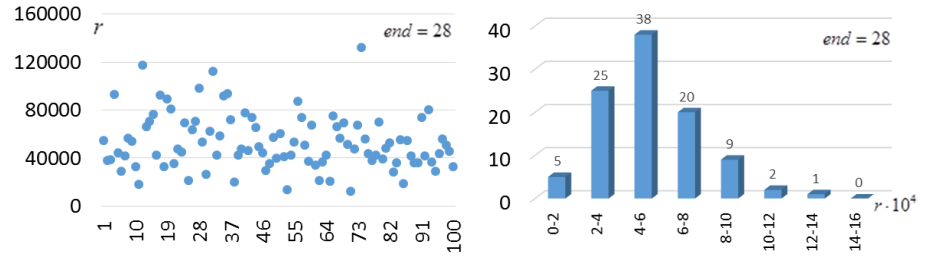
а *б*

Рис. 8. Результати тестування алгоритму Hill climbing з функцією вартості WCF, $end = 20$



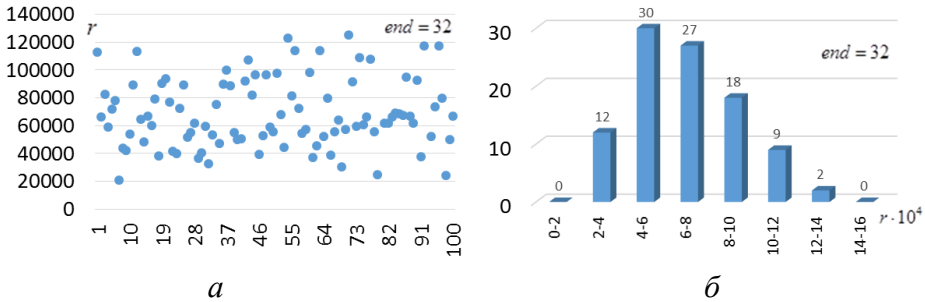
а *б*

Рис. 9. Результати тестування алгоритму Hill climbing з функцією вартості WCF, $end = 24$



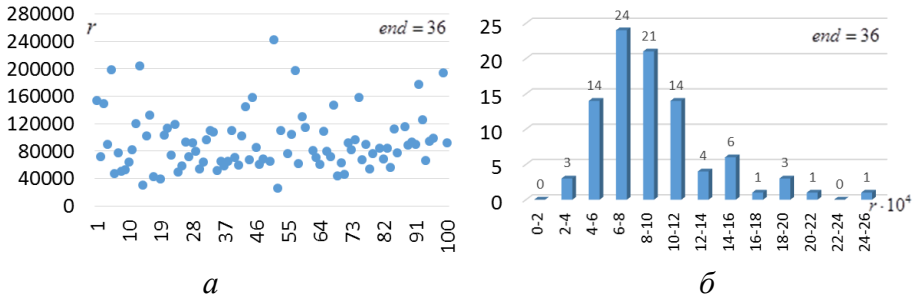
а *б*

Рис. 10. Результати тестування алгоритму Hill climbing з функцією вартості WCF, $end = 28$



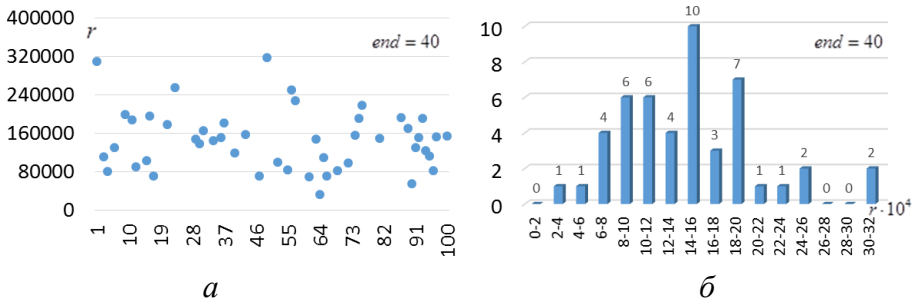
а *б*

Рис. 11. Результати тестування алгоритму Hill climbing з функцією вартості WCF, $end = 32$



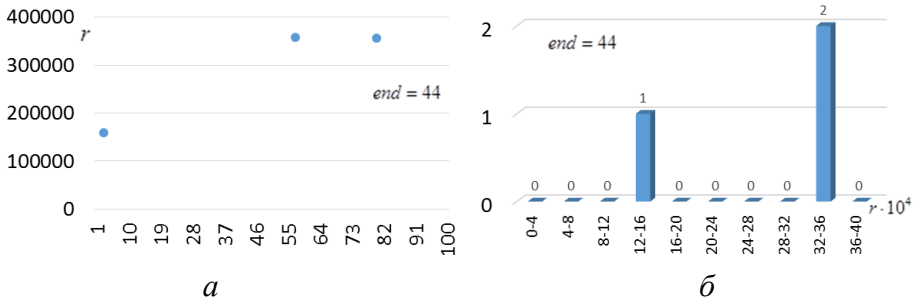
а *б*

Рис. 12. Результати тестування алгоритму Hill climbing з функцією вартості WCF, $end = 36$



а *б*

Рис. 13. Результати тестування алгоритму Hill climbing з функцією вартості WCF, $end = 40$



а *б*

Рис. 14. Результати тестування алгоритму Hill climbing з функцією вартості WCF, $end = 44$

Таблиця 1

Узагальнені результати тестування алгоритму Hill climbing із функцією вартості WCF для різних значень *end*

end	Кількість знайдених S-блоків	Середня кількість ітерацій (r^{aver})
0	0	–
4	0	–
8	0	–
12	0	–
16	58	276 380
20	91	110 770
24	99	69 344
28	100	53 160
32	98	68 855
36	92	92 709
40	48	146 074
44	3	291 568
48	0	–

Обговорення результатів

За результатами тестування отримали суттєве прискорення при формуванні цільових підстановок. Зокрема, у порівнянні із кращим відомим на сьогоднішній день результатом було суттєво зменшено середню кількість ітерацій.

Значення $end = 32$ відповідає випадку функції вартості, яку було запропоновано та досліджено у [8, 9]. За результатом тестування алгоритму Hill climbing у [8] отримано середню кількість ітерацій $r^{aver} = 70{,}596$. В [9] тими ж авторами опубліковано кращий результат, який складає 65,933 ітерацій. Наша оцінка з табл. 1 для $end = 32$ дає середню кількість 68,855 ітерацій, що є близьким до значень із [8, 9]. Отже цей результат вважаємо підтвердженим. Однак з отриманих результатів (табл. 1) бачимо, що найменша кількість виконаних ітерації досягається при $end = 28$. При цьому, в середньому необхідно виконати лише 53,160 ітерації. Це майже на 20 % краще, ніж для результатів із [8, 9]. Крім того, ми значно покращили частоту формування цільових підстановок. За результатами тестування в [8] тільки в 11 випадках із 30 незалежних експериментів було знайдено S-блок з нелінійністю 104. Для наших налаштувань успіх було досягнуто в 100 % випадках.

Аналізуючи другий стовпчик в табл. 1, також бачимо, що для інших значень параметру end кількість знайдених S-блоків зменшується. Однак це відбувається не тому, що алгоритм не спроможний знайти рішення, а тому, що виконується умова виходу при досягненні обраного критерію зупинки у max_frozen_outer_loops=5. На рис. 15 та 16 наведено приклади послідовної зміни значень функції вартості WCF та відповідні значення нелінійності $N(S)$ для $end = 28$ і $end = 40$ під час роботи алгоритму пошуку. Як бачимо, в середньому, під час роботи алгоритму пошуку виконується від 100 до 150 покращень значень функції вартості WCF.

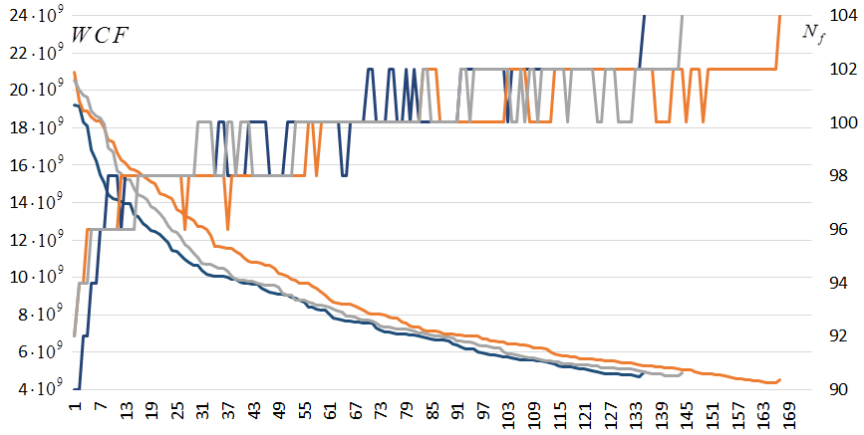


Рис. 15. Послідовність зміни значень WCF та $N(S)$, які фіксувалися при кожному покращенні значенні функції вартості WCF, $end = 28$

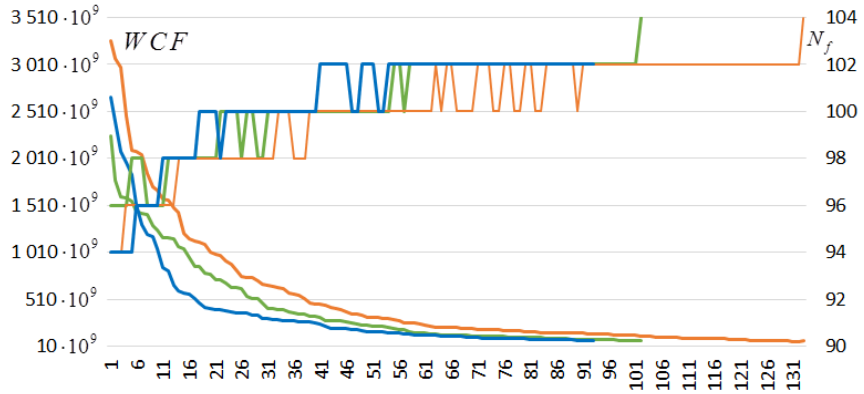


Рис. 16. Послідовність зміни значень WCF та $N(S)$, які фіксувалися при кожному покращенні значенні функції вартості WCF, $end = 40$

Висновки

У цій роботі досліджено можливість формування високонелінійних S-блоків за допомогою простого за реалізацією евристичного алгоритму сходження на пагорб. У якості функції вартості ми застосовували *WCF*.

За результатами тестування перевірено та підтверджено результати, які опубліковано у [8, 9]. Зокрема підтверджуємо, що функція вартості *WCF* дійсно дозволяє значно прискорити формування високонелінійних підстановок. При застосуванні алгоритму Hill climbing нами отримано середнє значення кількості ітерацій 68,855, що є близьким до опублікованих у [8, 9] результатів (65,933 та 70,596 відповідно).

Слід зазначити, що в [8, 9] застосовувався фіксований параметр $end = 32$. В цій роботі було модифіковано функцію вартості *WCF* та проведено розширене тестування для різних значень end. За результатами тестування кращим параметрам було визначено значення $end = 28$, при якому у 100 % (з проведених випробувань) було знайдено бієктивний S-блок з нелінійністю 104. При цьому середня кількість ітерацій алгоритму Hill climbing складала 53 160 ітерацій. Це майже на 20 % покращує відомий із [8, 9] результат.

Список літератури:

1. Freyre Echevarría A. Evolución híbrida de s-cajas no lineales resistentes a ataques de potencia. 2020.
2. McLaughlin J. Applications of search techniques to cryptanalysis and the construction of cipher components: phd. University of York, 2012.
3. Álvarez-Cubero J. Vector Boolean Functions: applications in symmetric cryptography. 2015.
4. Bard G.V. Algebraic Cryptanalysis. Boston, MA: Springer US, 2009.
5. Courtois N.T., Bard G.V. Algebraic Cryptanalysis of the Data Encryption Standard // Cryptography and Coding / ed. Galbraith S.D. Berlin, Heidelberg: Springer, 2007. P. 152–169.
6. Rodinko M., Oliynykov R., Gorbenko Y. Optimization of the High Nonlinear S-Boxes Generation Method // Tatra Mountains Mathematical Publications. Sciendo, 2017. Vol. 70, № 1. P. 93–105.
7. Kuznetsov A.A. et al. Stream Ciphers in Modern Real-time IT Systems. Cham: Springer Nature, 2022. 593 p.
8. Freyre-Echevarría A. et al. An External Parameter Independent Novel Cost Function for Evolving Bijective Substitution-Boxes: 11 // Symmetry. Multidisciplinary Digital Publishing Institute, 2020. Vol. 12, № 11. P. 1896.
9. Freyre Echevarría A., Martínez Díaz I. A new cost function to improve nonlinearity of bijective S-boxes. 2020.
10. Kuznetsov A. et al. Optimizing the Local Search Algorithm for Generating S-Boxes // 2021 IEEE 8th International Conference on Problems of Infocommunications, Science and Technology (PIC S T). 2021. P. 458–464.
11. Burnett L.D. Heuristic Optimization of Boolean Functions and Substitution Boxes for Cryptography: phd. Queensland University of Technology, 2005.
12. Ivanov G., Nikolov N., Nikova S. Cryptographically Strong S-Boxes Generated by Modified Immune Algorithm // Cryptography and Information Security in the Balkans / ed. Pasalic E., Knudsen L.R. Cham: Springer International Publishing, 2016. P. 31–42.
13. Freyre-Echevarría A. et al. Evolving Nonlinear S-Boxes With Improved Theoretical Resilience to Power Attacks // IEEE Access. 2020. Vol. 8. P. 202728–202737.
14. Kazymyrov O., Kazymyrova V., Oliynykov R. A Method For Generation Of High-Nonlinear S-Boxes Based On Gradient Descent: 578. 2013.
15. Clark J.A., Jacob J.L., Stepney S. The design of S-boxes by simulated annealing // New Gener Comput. 2005. Vol. 23, № 3. P. 219–231.
16. McLaughlin J. Applications of search techniques to cryptanalysis and the construction of cipher components: phd. University of York, 2012.
17. Wang J. et al. Construction Method and Performance Analysis of Chaotic S-Box Based on a Memorable Simulated Annealing Algorithm: 12 // Symmetry. Multidisciplinary Digital Publishing Institute, 2020. Vol. 12, № 12. P. 2115.
18. Kuznetsov A. et al. WHS Cost Function for Generating S-boxes // 2021 IEEE 8th International Conference on Problems of Infocommunications, Science and Technology (PIC S T). 2021. P. 434–438.
19. Ivanov G., Nikolov N., Nikova S. Reversed genetic algorithms for generation of bijective s-boxes with good cryptographic properties // Cryptogr. Commun. 2016. Vol. 8, № 2. P. 247–276.
20. Kapuściński T., Nowicki R.K., Napoli C. Application of Genetic Algorithms in the Construction of Invertible Substitution Boxes // Artificial Intelligence and Soft Computing / ed. Rutkowski L. et al. Cham: Springer International Publishing, 2016. P. 380–391.

21. Mariot L., Leporati A. Heuristic Search by Particle Swarm Optimization of Boolean Functions for Cryptographic Applications // Proceedings of the Companion Publication of the 2015 Annual Conference on Genetic and Evolutionary Computation. New York, NY, USA: Association for Computing Machinery, 2015. P. 1425–1426.
22. Picek S., Cupic M., Rotim L. A New Cost Function for Evolution of S-Boxes // Evolutionary Computation. 2016. Vol. 24, № 4. P. 695–718.
23. Clark A.J. Optimisation heuristics for cryptology: phd. Queensland University of Technology, 1998.
24. Cusick T., Stănică P. Cryptographic Boolean Functions and Applications: Second edition // Cryptographic Boolean Functions and Applications: Second Edition. 2017. P. 2751 p.

Надійшла до редколегії 05.03.2022

Відомості про авторів:

Кузнецов Олександр Олександрович – д-р техн. наук, професор, Харківський національний університет імені В.Н. Каразіна, професор кафедри безпеки інформаційних систем і технологій, факультет комп'ютерних наук; Україна; e-mail: kuznetsov@karazin.ua, ORCID: https://orcid.org/0000-0003-2331-6326

Горбенко Юрій Іванович – канд. техн. наук, Харківський національний університет імені В.Н. Каразіна, старший науковий співробітник кафедри безпеки інформаційних систем і технологій, факультет комп'ютерних наук; Україна; e-mail: gorbenkou@iit.kharkov.ua, ORCID: https://orcid.org/0000-0002-0652-8629

Полуяненко Микола Олександрович – канд. техн. наук, Харківський національний університет імені В.Н. Каразіна, доцент кафедри безпеки інформаційних систем і технологій, факультет комп'ютерних наук; Україна; e-mail: nlfsr01@gmail.com, ORCID: https://orcid.org/0000-0001-9386-2547

Кандій Сергій Олегович – АТ «Інститут інформаційних технологій», технік-конструктор, Україна; e-mail: sergeykandy@gmail.com, ORCID: https://orcid.org/0000-0003-0552-8341

Матвєєва Євгенія Дмитрівна – Харківський національний університет імені В.Н. Каразіна, студент кафедри безпеки інформаційних систем і технологій, факультет комп'ютерних наук; Україна; e-mail: belka.j.0507@gmail.com, ORCID: https://orcid.org/0000-0001-9834-2970