**УДК 004.056.5** **DOI:10.30837/rt.2022.2.209.10**

*О.О. КУЗНЕЦОВ, д-р техн. наук, М.О ПОЛУЯНЕНКО, д-р техн. наук,*
*С.О. КАНДІЙ, Є.О. ЛОГАЧОВА*

# ОБҐРУНТУВАННЯ ПАРАМЕТРІВ АЛГОРИТМУ ІМІТАЦІЇ ВІДПАЛУ ДЛЯ ПОШУКУ НЕЛІНІЙНИХ ПІДСТАНОВОК СИМЕТРИЧНИХ ШИФРІВ

## Вступ

Нелінійні підстановки (S-boxes) відігріють вирішальну роль у забезпеченні певних криптографічних властивостей сучасних шифрів із секретним ключем [1 – 3]. Зокрема, вони забезпечують confusion та вносять нелінійність у зв'язок вхід-вихід шифру. Отже генерація S-boxes із потрібними криптографічними властивостями є важливим завданням.

В цій роботі розглядаємо генерацію випадкових 8-бітних бієктивних підстановок, які зазвичай застосовуються в сучасних алгоритмах шифрування із симетричним ключем [4 – 6]. Досліджуємо алгоритм імітації відпалу (SA) та оптимізуємо його параметри за критерієм мінімізації обчислювальних ресурсів на генерацію цільового S-box. Зокрема досліджуємо вплив початкової температури та «коефіцієнта охолодження» на ефективність пошуку. Далі показуємо, що при обранні рекомендованих параметрів ймовірність знайти цільовий S-блок дорівнює 56,4 % а середній час пошуку складає 14,2 с. Це кращий відомий результат при застосуванні алгоритму імітації відпалу.

## Пов’язані роботи

Методи імітації відпалу досліджуються в різних практичних застосуваннях математичної оптимізації [7]. В роботі [8] було запропоновано реалізацію цього алгоритму для пошуку нелінійних підстановок. Також в цій роботі запропоновано функцію вартості based on Walsh-Hadamard Spectra (WHS). Згодом ця функція вартості досліджувалася в різних роботах, наприклад у [9 – 12] та багатьох інших. Слід відмітити, що у [8] авторам не вдалося сформувати S-блок із нелінійністю >102 через високу обчислювальну складність пошуку. Кращий результат, якого вони досягли, це підстановки із нелінійністю 102. В роботі [9] проведено дослідження налаштувань функції вартості WHS та сформовано S-блок із нелінійністю 104. Але для цього було застосовано інший алгоритм пошуку (використовувався новий алгоритм «Genetic and Tree»). Складність пошуку також виявилася занадто високою (понад 3 мільйони ітерацій). Подальші дослідження були спрямовані на як на розробку нових функцій вартості [10, 12, 13], так і на застосування нових алгоритмів пошуку [3, 14 – 16]. Зокрема в роботах [17 – 20] досліджені різні варіанти алгоритму імітації відпалу. Однак в цих та інших роботах застосування імітації відпалу для генерації S-блоків виявилося надзвичайно складним. Наприклад, в [19] повідомлялося про генерацію бієктивних 8-бітних S-блоків із нелінійностю 104, але для цього виконувалися експерименти із понад 30,000,000 ітераціями. В роботі [18] також проведено чисельні експерименти, але авторам не вдалося сформувати підстановку із нелінійністю вище 100. В одній з останніх робіт [20] авторами досліджено нові методи на основі імітації відпалу. Але згенерований ними S-блок (наведений в статті) має нелінійність лише 92. Отже обчислювальна складність відомих застосувань методу імітації відпалу до задачі генерації нелінійних підстановок є дуже високою. В статті пропонуємо власну реалізацію алгоритму імітації відпалу та показуємо, що генерація бієктивних 8-бітних S-блоків із нелінійностю 104 виконується значно швидше.

## Методи

Алгоритм імітації відпалу відносять до класу ітеративних алгоритмів математичної оптимізації та до більш широкого класу евристичних методів рішення комбінаторних задач. Такі алгоритми використовують існуючу сукупність рішень (популяцію) для внесення деякі-

лькох поступових змін і створення (оновлення) нової популяції. Більш детально ці методи описані у [7].

Перше застосування алгоритму імітації відпалу до задачі генерації S-блоків наведено у [8]. В роботах [19, 21] наведено сучасну версію алгоритму (див. рис. 1), яку адаптовано до задачі пошуку нелінійних S-блоків. В цій роботі застосовуємо цей алгоритм з невеликими змінами. У базовому алгоритмі як критерій прийняття нового рішення $S_n$ застосовується покращення функції вартості $C(S_n)$:

$$C(S_n) < C(bestsol). \quad (1)$$

Ми приймаємо всі нові рішення, для яких функція вартості не погіршується:

$$C(S_n) \le C(bestsol). \quad (2)$$

Застосування критерію (2) замість (1) суттєво збільшує коло можливих рішень. Крім того, нами було виконано адаптацію роботи SA алгоритму у багатопоточному режимі пошуку.

$S \leftarrow S_0$
$bestsol \leftarrow S_0$
$T \leftarrow T_0$
$ZERO\_ACCEPT\_LOOPS \leftarrow 0$
**for** $x \leftarrow 0, MAX\_OUTER\_LOOPS - 1$ **do**
  $ACCEPTS\_IN\_THIS\_LOOP \leftarrow false$
  **for** $y \leftarrow 0, MAX\_INNER\_LOOPS - 1$ **do**
    Choose some $S_n$ in the 1-move neighbourhood of $S$.
    $cost\_diff \leftarrow C(S_n) - C(S)$
    **if** $cost\_diff < 0$ **then**
      $S \leftarrow S_n$
      $ACCEPTS\_IN\_THIS\_LOOP \leftarrow true$
      **if** $C(S_n) < C(bestsol)$ **then**
        $bestsol \leftarrow S_n$
      **end if**
    **else**
      $u \leftarrow Rnd(0, 1)$
      **if** $u < exp(-cost\_diff/T)$ **then**
        $S \leftarrow S_n$
        $ACCEPTS\_IN\_THIS\_LOOP \leftarrow true$
      **end if**
    **end if**
  **end for**
  **if** $ACCEPTS\_IN\_THIS\_LOOP = false$ **then**
    $ZERO\_ACCEPT\_LOOPS \leftarrow ZERO\_ACCEPT\_LOOPS + 1$
    **if** $ZERO\_ACCEPT\_LOOPS = MAX\_FROZEN\_OUTER\_LOOPS$ **then**
      ▷ Algorithm terminates early.
      **return** $bestsol$
    **end if**
  **end if**
  $T \leftarrow T \times \alpha$
**end for**
**return** $bestsol$

Рис. 1. Псевдокод алгоритму імітації відпалу з [19, 21]

В роботі розглядаємо лише випадок генерації 8-бітної бієктивної підстановки $S_n$. В якості функції вартості підстановки $S_n$ використовуємо функцію з [13, 16]:

$$\max WHS = \sum_{i=1}^{255} \left| \max\left(WHT\right) - X \right|^R, \tag{3}$$

де $WHT$ – спектральні коефіцієнти Уолша–Адамара (англ. Walsh–Hadamard transform);

- $X$ та $R$ – деякі параметрів цільової функції $WHS$.
- В якості оптимальних параметрів функції (3) обрано [13, 16]:
- $X = 36$ як максимально допустиме значення, яке зменшує $\max WHS$, але не приводить до суттєвого впливу на її адекватне взаємозв'язок з нелінійністю S-блоку;
- $R = 4$ як максимально допустиме значення, яке збільшує діапазон значень функції $\max WHS$, що може покращити «чутливість» алгоритмів формування S-блоків.

Зазначимо, що під час обчислення функції вартості $\max WHS$ одночасно розраховувалася нелінійність S-блоку:

$$N_f = \frac{1}{2} \cdot \left(2^n - \max\left(WHT\right)\right) = 128 - \frac{1}{2} \cdot \max\left(WHT\right). \tag{4}$$

Для зниження температури з плином часу застосовувався коефіцієнт охолодження $\alpha$.

Пошук починається з деякого випадково згенерованого S-блоку $S_0$. Процес пошуку розбивається на два цикли: зовнішній та внутрішній.

Зовнішній цикл виконується до того моменту, коли один із згенерованих S-блоків $S_n$ буде відповідати заданим параметрам або виконується інший критерій зупинки.

У внутрішньому циклі на кожній ітерації генерується деяке поточне рішення шляхом випадкової перестановки двох значень S-блоку. Далі розраховується функція вартості для поточного рішення та порівнюється з вже найденим кращим рішенням. Якщо значення не гірше попереднього кращого S-блоку, то попереднє краще рішення замінюється поточним. Якщо значення є гіршим, тоді в залежності від температури з деякою ймовірністю приймається гірше рішення.

**Умови тестування**

При реалізації алгоритму імітації відпалу для генерації S-блоків ми застосовували наступні вихідні параметри:

- COST_FUNCTIONS – функція вартості, яка застосовується в алгоритмі пошуку. В цій статті застосовували COST_FUNCTIONS = $\max WHS$;
- THREADS_COUN – кількість потоків, у яких проходить одночасний пошук. В нашому випадку THREADS_COUN =30, що відповідало максимальної кількості потоків, який підтримував процесор комп'ютера;
- $T_0$ – початкове значення "температури". У [21] зазначено, що $T_0$ повинна забезпечувати початкову ймовірність прийняття гіршого рішення на рівні 50 – 80 %. В роботі досліджуємо ефективність пошуку при різних значеннях $T_0$;
- $\alpha$ – «коефіцієнт охолодження», який визначає, наскільки температура знижується на кожній ітерації алгоритму. В роботі досліджуємо ефективність пошуку при різних значеннях $\alpha$;
- MAX_INNER_LOOPS, що визначає кількість внутрішніх циклів, які може здійснити локальний алгоритм пошуку при кожній температурі. В статті застосовували MAX_INNER_LOOPS=650 (тобто, загальна кількість внутрішніх тестувань становила $30 \cdot 650 = 19\ 500$);
- Критерії зупинки. В якості критеріїв зупинки використовувались наступні:

- $N_f$ – цільове значення нелінійності (3) S-блоку. В наших експериментах обмежилися значенням $N_f = 104$, тобто пошук припиняється коли знайдено $S_n$ із нелінійністю 104;
- MAX_OUTER_LOOPS – максимальна кількість зовнішніх циклів, тобто скільки разів SA алгоритму дозволялося знижувати температуру та продовжувати пошук до того, як він зупиниться. В дослідженнях застосовували MAX_OUTER_LOOPS =50;
- MAX_FROZEN_OUTER_LOOPS – кількість поспіль виконаних зовнішніх циклів при яких не знайдено жодного покращення функції вартості. В роботі ми застосовували MAX_FROZEN_OUTER_LOOPS = 5.

Окремі параметри алгоритму (MAX_INNER_LOOPS, MAX_OUTER_LOOPS, MAX_FROZEN_OUTER_LOOPS) обирались з міркувань, наведених у [16].

Початкова температура змінювалась від значення, де ймовірність прийняття гіршого рішення майже дорівнювала нулю до більш високої, де ймовірність становила близькою до одиниці.

Збільшення $T_0$ проводилось за правилом

$$T_0^{i+1} = 1{,}13 \cdot T_0^i . \tag{5}$$

Для кожного $T_0^i$ з (5) проведено 100 запусків алгоритму імітації відпалу.

Параметр $\alpha$ змінювався від 0,6 до 0,95:

- для $\alpha = 0{,}6$ було проведено 10 100 запусків алгоритму пошуку;
- для $\alpha = 0{,}7$ було проведено 7 600 запусків;
- для $\alpha = 0{,}8$ було проведено 5 700 запусків;
- для $\alpha = 0{,}9$ було проведено 8 200 запусків;
- для $\alpha = 0{,}95$ було проведено 8 200 запусків.

Обмеження max_frozen_outer_loops=5 призводило до втрачання частки рішень, щодо яких алгоритм ще міг знайти цільовий S-блок. Але доцільність подальшого пошуку вважалась за малу у порівнянні з затраченим часом.

**Отримані результати**

Під час тестування проводилась оцінка кількості запусків SA алгоритму за максимальним значенням кортежу, протягом якого були відсутні покращення функції вартості у зовнішньому циклі, але потім було знайдено краще рішення стану S-блоку. Нагадаємо, якщо зазначений кортеж перевищував значення max_frozen_outer_loops, то алгоритм пошуку припинявся.

Результати щодо відносної кількісті запусків алгоритмів пошуку в залежності від довжини максимальних кортежів для $\alpha = 0{,}6$; 0,7; 0,8 0,9 та 0,95, а також разом з абсолютними значеннями наведено у табл. 1. Для порівняння наведено аналогічні значення, які було отримано алгоритмом локального пошуку (рядок позначено при $\alpha$ символом « – ») [16].

Як бачимо, тенденція розподілу кортежів для алгоритму імітації відпалу для перших значень $\alpha$ зберігається так само як і для алгоритму локального пошуку. З ростом параметру $\alpha$ зростає кількість кортежів більшого розміру. Дане зростання пояснюється зростанням кількості прийнятих погіршень, що підвищує ймовірність виходу системи зі стану локального мінімуму.

Таблиця 1

Розподіл кількості запусків алгоритмів пошуку від максимальної кількості поспіль зовнішніх циклів (кортежу), при яких не знайдено жодного покращення, однак потім покращення мали місце

| $\alpha$ | Максимальна довжина кортежу | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 0 | | 1 | | 2 | | 3 | | 4 | | 5 | |
| – | 720 | 48% | 534 | 36% | 148 | 10% | 57 | 4% | 24 | 2% | 4 | 0,3% |
| 0,6 | 4 281 | 42% | 3 042 | 30% | 1 509 | 15% | 725 | 7% | 402 | 4% | 141 | 1,4% |
| 0,7 | 2 722 | 36% | 2 510 | 33% | 1 245 | 16% | 641 | 8% | 355 | 5% | 127 | 1,7% |
| 0,8 | 1 362 | 24% | 2 060 | 36% | 1 244 | 22% | 592 | 10% | 322 | 6% | 120 | 2,1% |
| 0,9 | 2 168 | 26% | 1 671 | 20% | 1 634 | 20% | 1 466 | 18% | 955 | 12% | 306 | 3,7% |
| 0,95 | 2 299 | 28% | 1 684 | 21% | 1 333 | 16% | 1 163 | 14% | 1 167 | 14% | 554 | 6,8% |

Основним з показників ефективності алгоритму пошуку можна вважати ймовірність формування цільового S-блоку. Відповідні результати тестування наведені на рис. 2 та 3 (відповідно для $\alpha = 0{,}6$ та 0,9). Ймовірність обчислювалась як відношення кількості знайдених цільових S-блоків до загальної кількості запусків алгоритму пошуку. Також вимірювали середній час формування S-блоку. Отримані результати наведено на рис. 4 та 5 (відповідно для $\alpha = 0{,}6$ та 0,9). Середній час обчислювався як інтервал часу між початком пошуку та знаходженням цільового S-блоку включаючи час затрачений на невдалі запуски алгоритму пошуку.

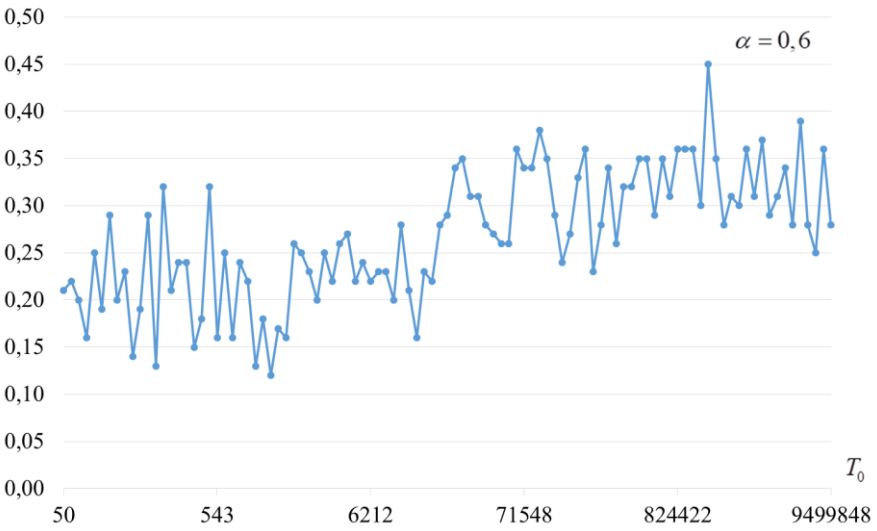


Рис. 2. Ймовірність формування цільового S-блоку при $\alpha = 0{,}6$

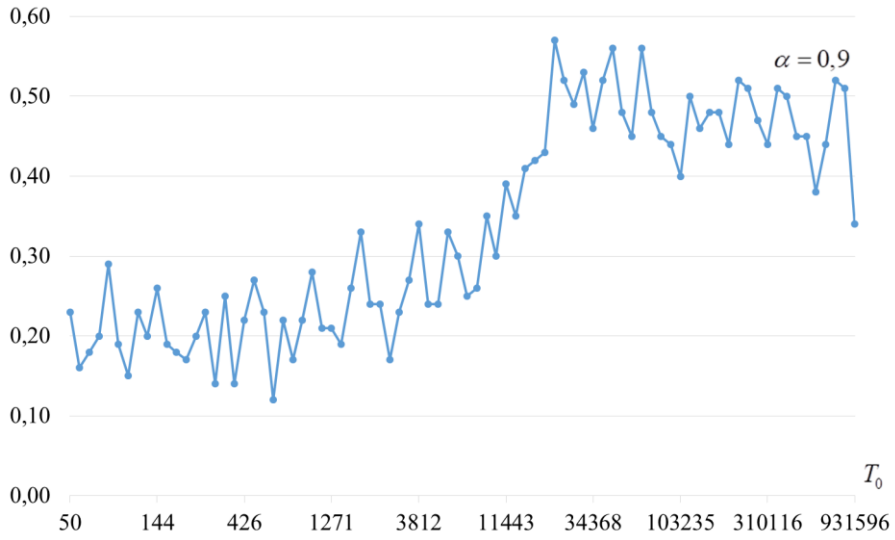


Рис. 3. Ймовірність формування цільового S-блоку при $\alpha = 0,9$

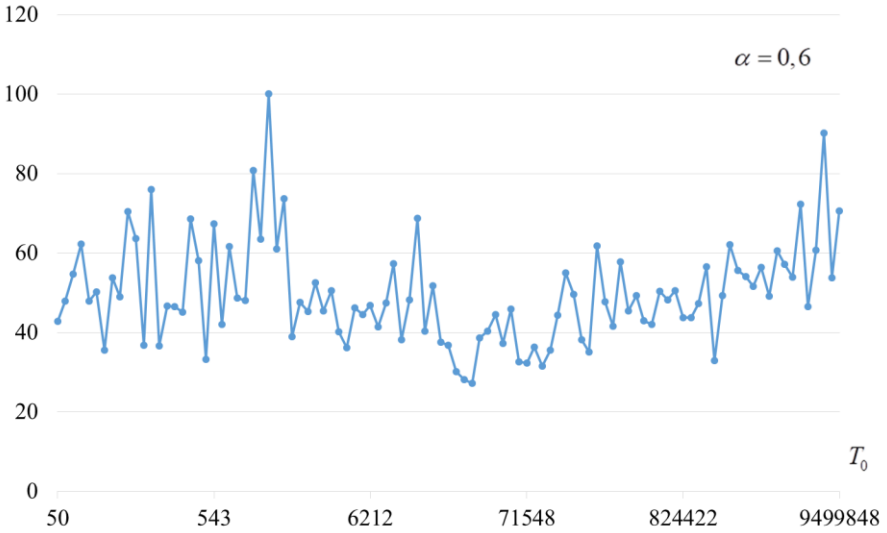


Рис. 4. Середній час (с) формування цільового S-блоку при $\alpha = 0,6$

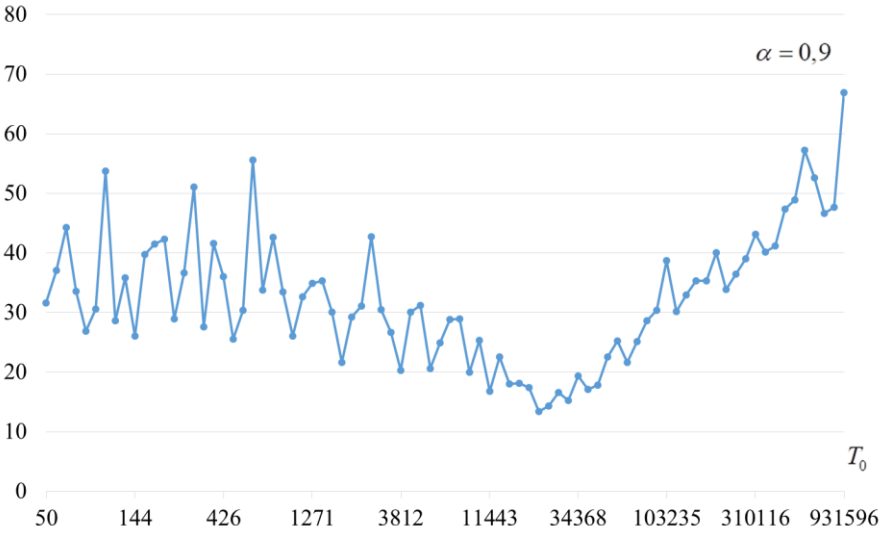


Рис. 5. Середній час (с) формування цільового S-блоку при $\alpha = 0,9$

При аналізу середнього часу необхідно обумовити, що обчислювання проходили на двох різних комп'ютерах, тому більш коректно аналізувати загальний характер поведінки зміни часу, ніж абсолютні значення.

Умовно, графіки ймовірності формування цільового S-блоку можливо поділити на три частини: ліва, середня та права.

- Ліва частина (від $T_0 = 50$ до 6 000), відповідає частині графіку, де значення ймовірності майже не змінюється або спостерігається невелике зростання.
- Середня частина ( $T_0 = 6\,000 \ldots 50\,000$ ), де спостерігається значний ріст ймовірності формування цільового S-блоку зі зростанням значення $T_0$. Зі збільшенням параметру $\alpha$ ріст відбувається більш швидко.
- Права частина (від $T_0 = 50\,000$ та вище), де, як і у лівій частини, ймовірність фіксується у деякого значення (зі збільшенням $\alpha$ це значення також збільшується), після чого майже не змінюється.

Аналогічним чином можливо умовно розділити графік середнього часу формування цільового S-блоку на дві частини.

- Ліва частина відповідає значенням $T_0$ від мінімальних до 20 000 – 40 000. Вона відповідає поступовому зниженню середнього часу пошуку цільового S-блоку. Значення $T_0 = 20\,000$ – 40 000 відповідає досягненням області мінімальних значень часу пошуку.
- Права частина – від $T_0 = 20\,000$ –. 40 000 та вище, характеризується зростанням середнього часу пошуку цільового S-блоку, при збільшені значення $\alpha$ швидкість зростання також збільшується.

Значною мірою на час пошуку впливає кількість виконаних ітерації зовнішнього циклу.

На рис. 6 та 7 (відповідно для $\alpha = 0{,}6$ та 0,9) наведена кількість ітерацій зовнішнього циклу до виконання умов зупинки з усієї множини запусків алгоритму для фіксованого значення $T_0$. Верхня крива відповідає максимальному значенню, нижня крива – мінімальному значенню, середня – середній арифметичній кількості ітерацій зовнішнього циклу з усіх запусків при фіксованому $T_0$.

На рис. 8 та 9 (відповідно для $\alpha = 0{,}6$ та 0,9) аналогічні показники, але лише з тих запусків, у яких було знайдено цільовий S-блок.

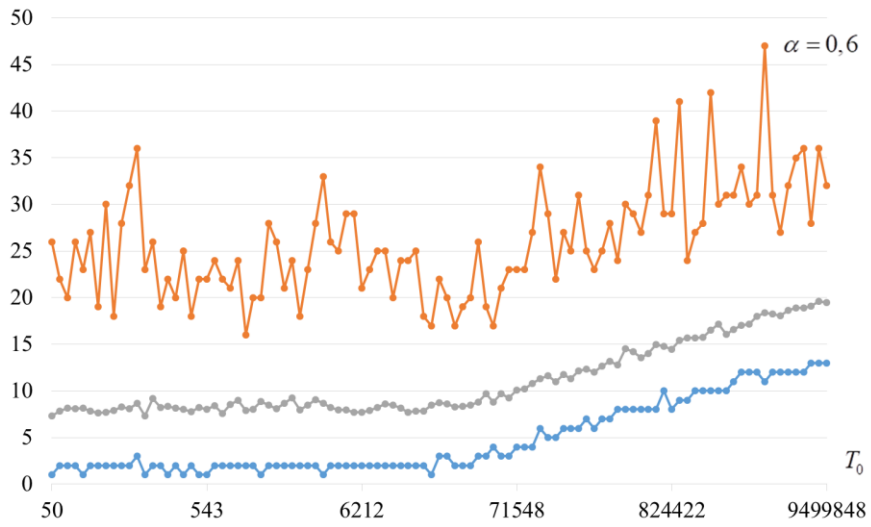


Рис. 6. Максимальна (верхня крива), середньоарифметична (посередині), мінімальна (нижня крива) кількості ітерацій зовнішнього циклу до виконання умов зупинки при $\alpha = 0{,}6$

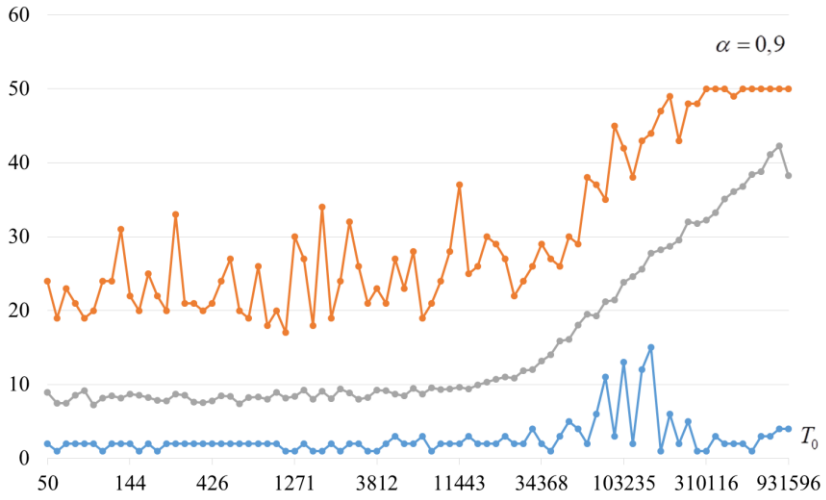


Рис. 7. Максимальна (верхня крива), середньоарифметична (посередині), мінімальна (нижня крива) кількості ітерацій зовнішнього циклу до виконання умов зупинки при $\alpha = 0,9$

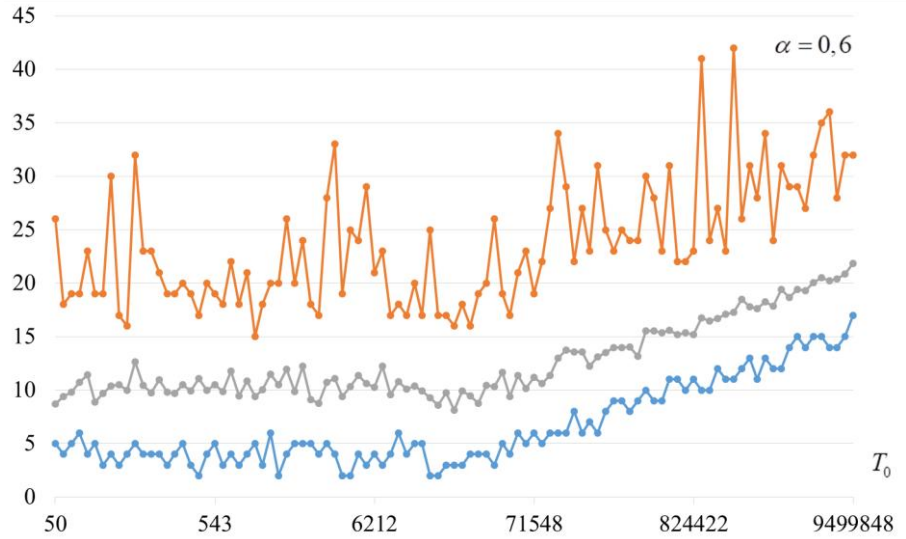


Рис. 8. Максимальна (верхня крива), середньоарифметична (посередині), мінімальна (нижня крива) кількості ітерацій зовнішнього циклу, за умови знайденого цільового S-блоку, до виконання умов зупинки при $\alpha = 0,6$

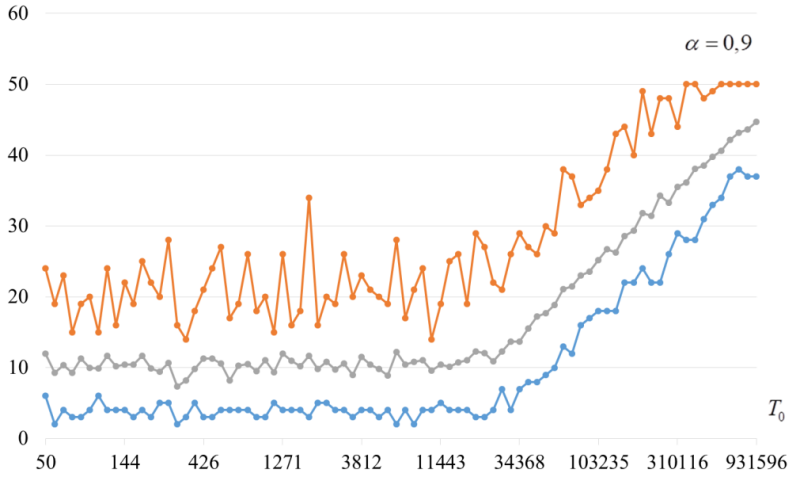


Рис. 9. Максимальна (верхня крива), середньоарифметична (посередині), мінімальна (нижня крива) кількості ітерацій зовнішнього циклу, за умови знайденого цільового S-блоку, до виконання умов зупинки при $\alpha = 0,9$

Поведінка кривих на наведених графіках відповідає умовним частинам графіків середнього часу формування цільового S-блоку. Ліва частина характеризується майже повною відсутністю зростання кількості ітерацій зовнішнього циклу до виконання умов зупинки. Після відмітки у $T_0 = 20\,000 - 40\,000$ спостерігається зростання у лінійному виді, з коефіцієнтом нахилу пропорційному параметру $\alpha$.

Більш докладно дослідити процеси, які відбуваються у системі під час виконання пошуку, можна за допомогою дослідження окремих подій. Будемо досліджувати такі події як зміни нелінійності $N_f$ та кількості прийнятих окремих рішень при кожному значенні $T_0$ та у кінці кожної ітерації зовнішнього циклу.

Прогрес зміни значення $N_f$ будемо оцінювати як середньоарифметичне значення $N_f$ всіх поточних рішень які тестуються при пошуку. Приклад прогресу зміни значення $N_f$ наведено на тривимірних графіках рис. 10.

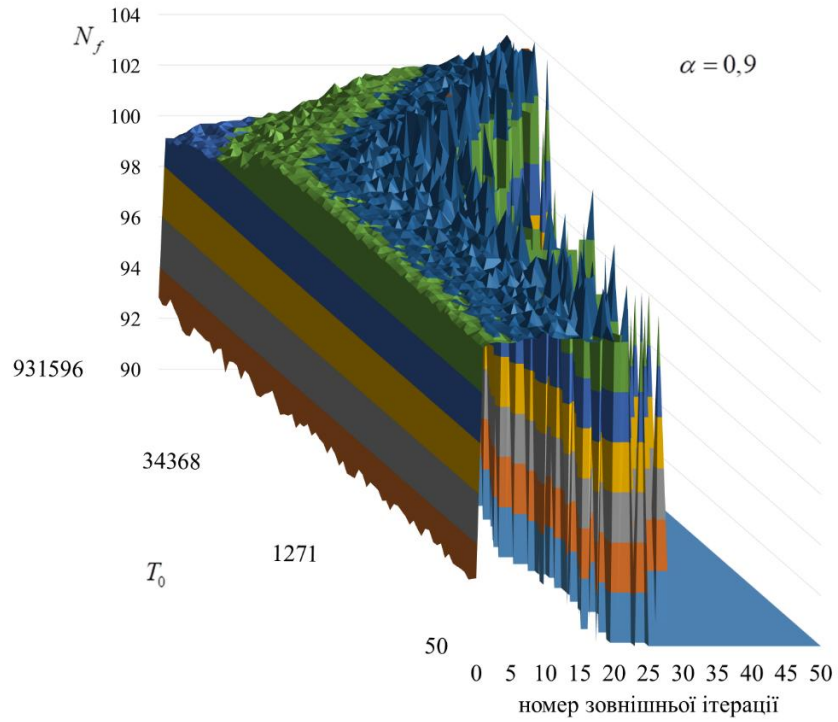


Рис. 10. Середньоарифметичне значення $N_f$ у ітераціях при $\alpha = 0,9$

Середньоарифметичне значення $N_f$ на початку роботи алгоритму відповідає випадково сформованим S-блокам (на графіку їх приведено під нульовою ітерацією зовнішнього циклу). Зі збільшенням номеру ітерації зовнішнього циклу збільшується й середньоарифметичне значення $N_f$. Однак темп збільшення відбувається по-різному в різних частинах графіку. Додержуючись раніше введеного умовного поділу графіку на ліву (при $T_0$ від мінімальних значень до 20 000 – 40 000) та праву (при $T_0$ від 20 000 – 40 000 та максимальних значень) частини, можна охарактеризувати темп збільшення середньоарифметичного значення $N_f$. У лівій частині після першої ітерації зовнішнього циклу середньоарифметичне значення $N_f$ складає близько 101,8 та у продовж ще 15 – 25 циклів досягає значення 104 (або досягає інших критеріїв зупинки). У правій частині графіку зростання спостерігається значно повільніше. Наприклад, при $\alpha = 0,9$ і $T_0 = 931\,596$ після першої ітерації зовнішнього циклу

середньоарифметичне значення $N_f$ дорівнює 99,1 та лінійно підвищується до 101,8 лише на 34 ітерації (що відповідає температурі $T = 931\,596 \cdot 0{,}9^{34} \approx 25\,910$).

Зі збільшенням номеру ітерації зовнішнього циклу зменшується загальна кількість S-блоків, що перевіряється (S-блоки, які досягли критеріїв зупинки алгоритму, далі в пошуку участь не беруть), тому зі зменшенням загальної кількості S-блоків підвищується девіація значень.

Приклад для випадку $\alpha = 0{,}9$ зміни загальної кількості ітерації, яка складається з кількості ітерації у внутрішньому циклі та кількість виконаних ітерації зовнішнього циклу, наведена на рис. 11.

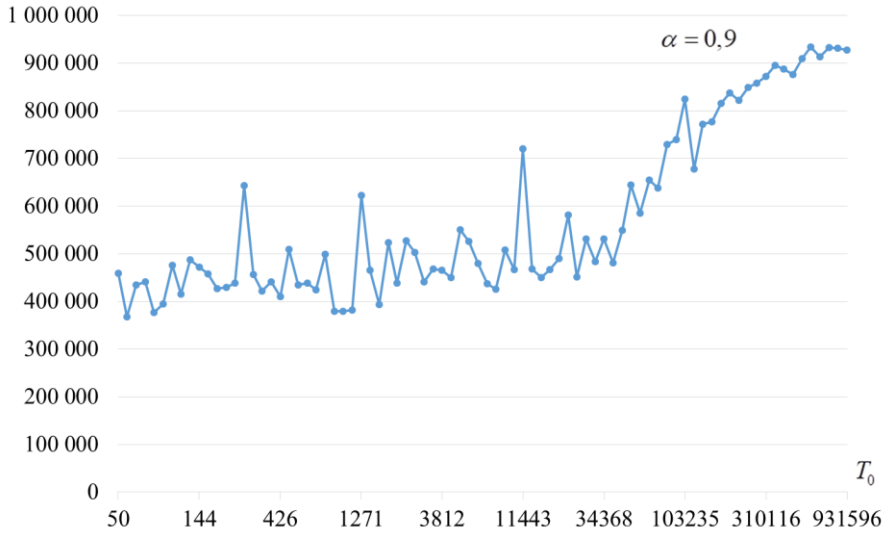


Рис. 11. Сумарне число середньоарифметичної кількості ітерації при пошуку цільового S-блоку при $\alpha = 0{,}9$

З ростом значення $T_0$ загальна кількість ітерації майже не змінюється до деякого значення ($T_0 = 30\,000 \ldots 70\,000$, в залежності від $\alpha$) та потім починає швидко зростати, що обумовлює значне підвищення часу на виконання кожного запуску алгоритму.

Для з'ясування факторів, що впливають на підвищення загальної кількості ітерації, розглянемо окремі події, що відбуваються у процесі виконання алгоритму:

- Середньоарифметичне значення (усереднене по 100 запускам) кількості прийнятих погіршень функції вартості, за окремий запуск алгоритму імітації відпалу, зображено на тривимірному графіку – рис. 12. На рис. 13 наведено сумарне значення середньоарифметичної кількості прийнятих погіршень функції вартості за всіма ітераціями впродовж одного запуску алгоритму. Двовимірний графік отримано з тривимірного шляхом підсумовування всіх значень у кожній ітерації для окремого значення температури;
- Приклад середньоарифметичного значення (усереднене по 100 запускам) кількості не прийнятих погіршень функції вартості, за окремий запуск, зображено на тривимірному графіку – рис. 14. На рис. 15 наведено сумарне значення середньоарифметичної кількості неприйнятих погіршень функції вартості за всіма ітераціями впродовж одного запуску алгоритму. Двовимірний графік отримано з тривимірного шляхом підсумовування всіх значень у кожній ітерації для окремого значення температури. Стан найкращого з найдених поточних рішень в обох випадках залишається незмінним;
- Середньоарифметичне значення (усереднене по 100 запускам) кількості прийнятих покращень функції вартості, за окремий запуск алгоритму, зображено на три-

вимірному графіку – рис. 16. На рис. 17 наведено сумарне значення середньоарифметичної кількості прийнятих покращень функції вартості за всіма ітераціями впродовж одного запуску алгоритму. Двовимірний графік отримано з тривимірного шляхом підсумовування всіх значень у кожній ітерації для окремого значення температури;

- Середнєьоарифметичне значення (усереднене по 100 запускам) кількості неприйнятих покращень функції вартості, за окремий запуск алгоритму, зображено на тривимірному графіку – рис. 18. На рис. 19 наведено сумарне значення середньоарифметичної кількості неприйнятих покращень функції вартості за всіма ітераціями впродовж одного запуску алгоритму. Двовимірний графік отримано з тривимірного шляхом підсумовування всіх значень у кожній ітерації для окремого значення температури.

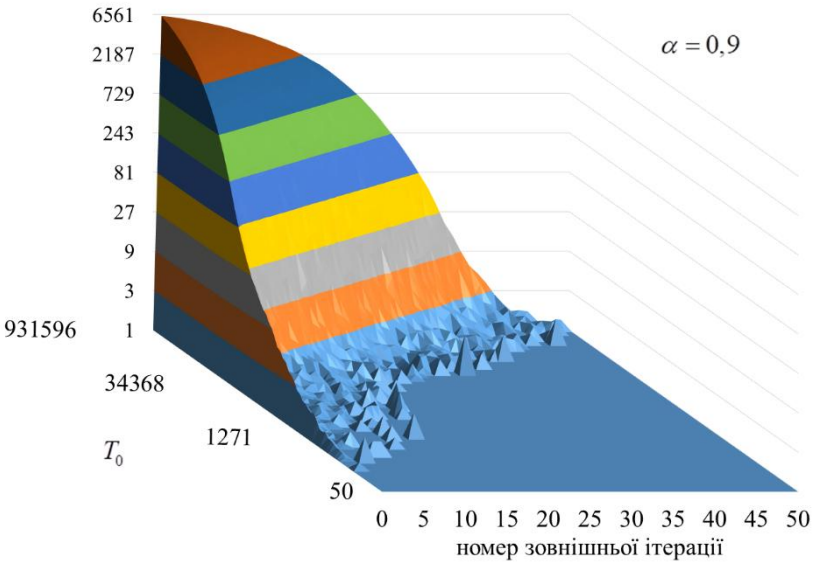


Рис. 12. Середньоарифметичне значення кількості прийнятих погіршень функції вартості у ітераціях при $\alpha = 0,9$

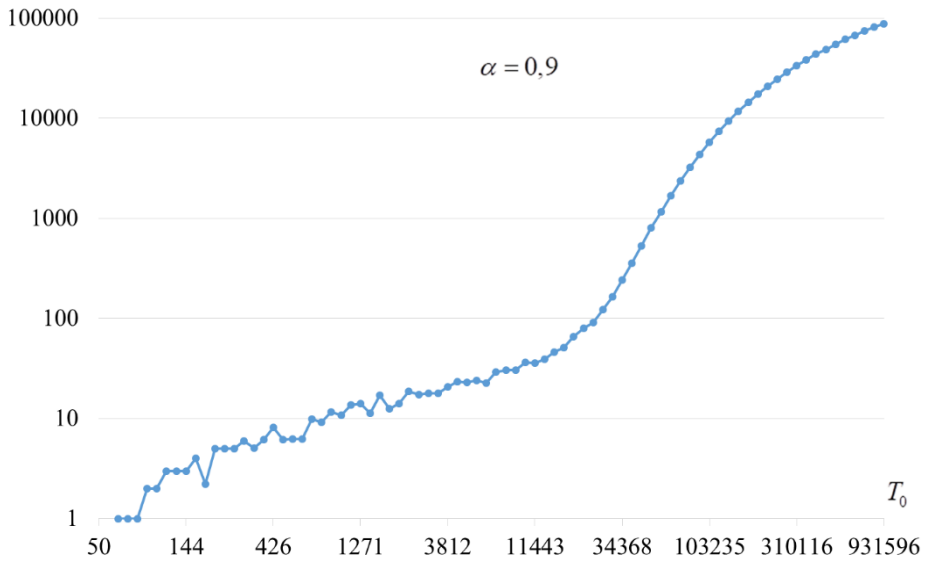


Рис. 13. Сумарне число середньоарифметичної кількості прийнятих погіршень функції вартості у ітераціях при $\alpha = 0,9$

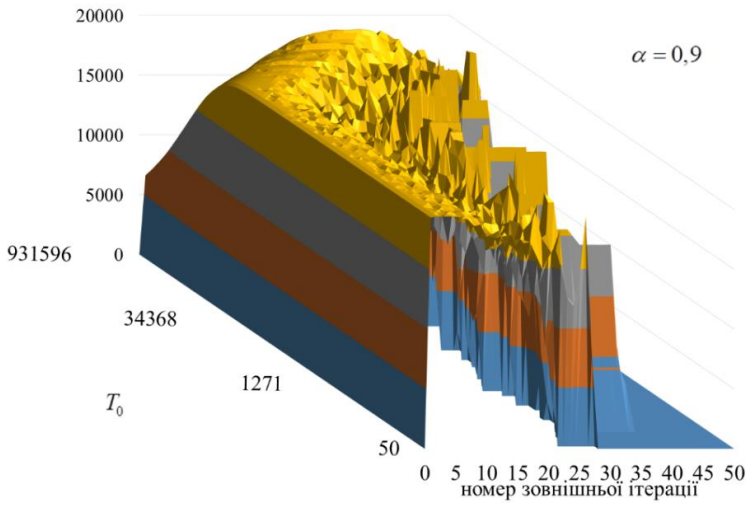


Рис. 14. Середньоарифметичне значення кількості неприйнятих погіршень функції вартості у ітераціях при $\alpha = 0,9$

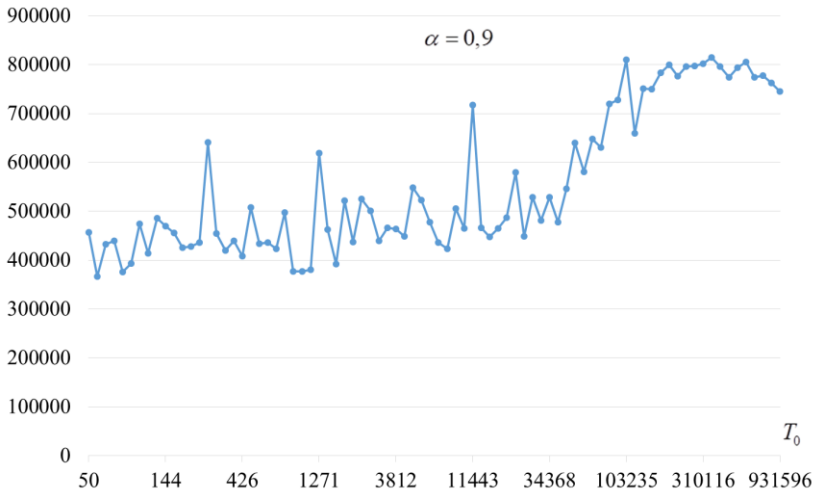


Рис. 15. Сумарне число середньоарифметичної кількості неприйнятих погіршень функції вартості у ітераціях при $\alpha = 0,9$

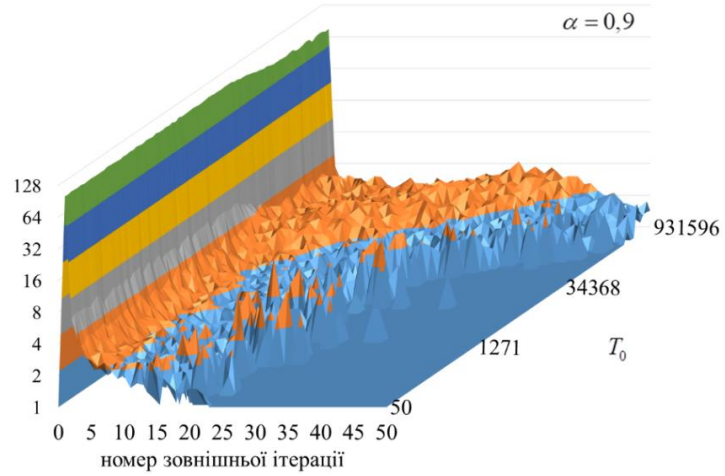


Рис. 16. Середньоарифметичне значення кількості прийнятих покращень функції вартості у ітераціях при $\alpha = 0,9$

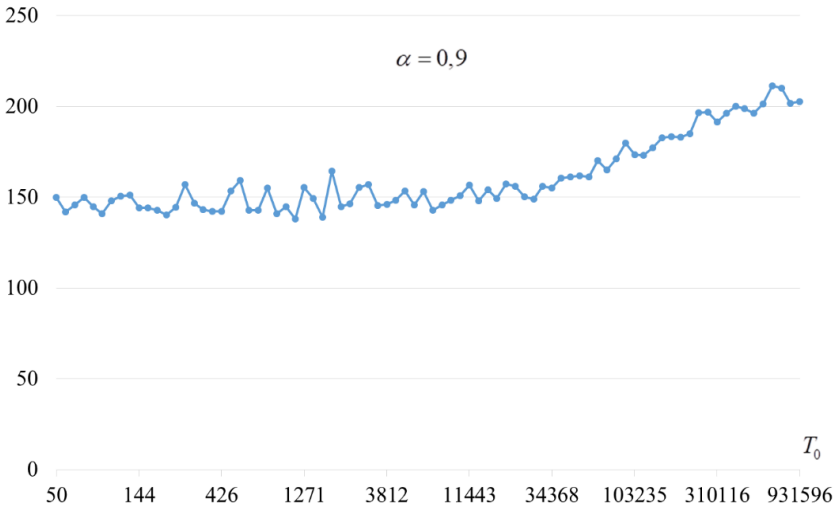


Рис. 17. Сумарне число середньоарифметичної кількості прийнятих покращень функції вартості у ітераціях при $\alpha = 0,9$

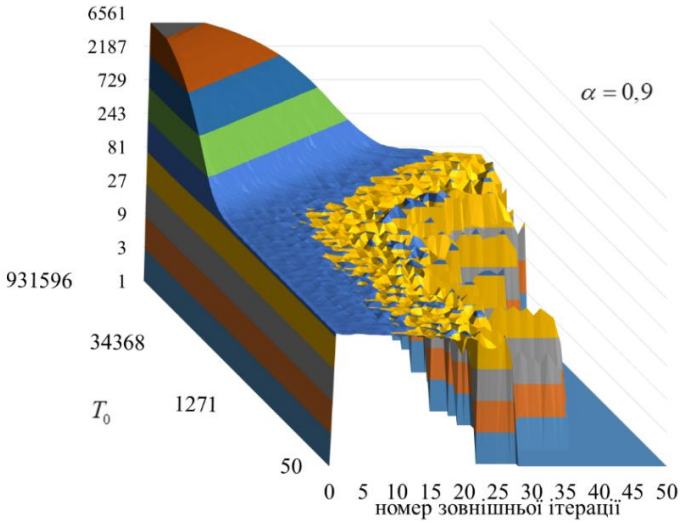


Рис. 18. Середньоарифметичне значення кількості неприйнятих покращень функції вартості у ітераціях при $\alpha = 0,9$

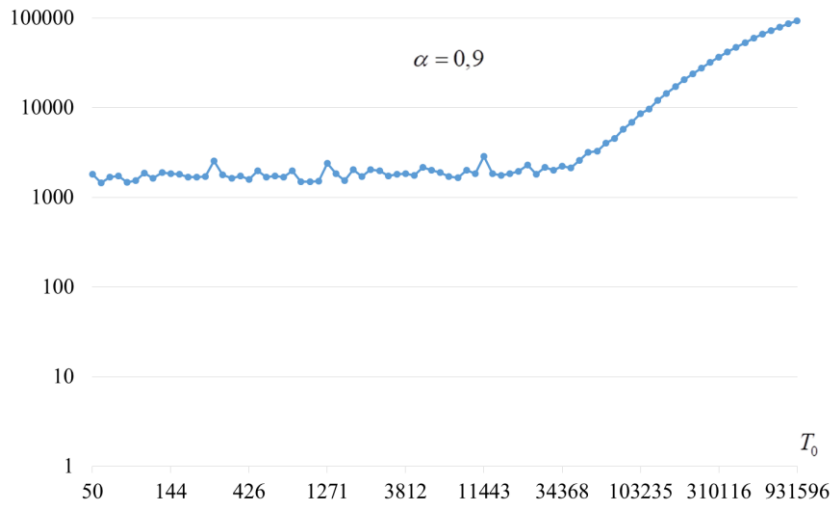


Рис. 19. Сумарне число середньоарифметичної кількості неприйнятих покращень функції вартості

**Обговорення результатів**

Як наведено у [16], при обраних вхідних параметрах, ймовірність формування цільового S-блоку за допомогою алгоритму локального пошуку складає 0,215, середній час пошуку одного блоку – 33,2 секунди.

При малих значеннях початкової температури ймовірність прийняття погіршуючого рішення дуже мала і тому алгоритм імітації відпалу веде себе яка звичайний алгоритм пошуку локального мінімуму та, відповідно, має такій ж самі значення ймовірності сформувати цільовий S-блок та середній час пошуку.

При збільшенні початкової температури збільшується ймовірність прийняття погіршуючих рішень, що приводить до виходу з поточного стану, яке, з одного боку, може бути непотрібним локальним мінімумом, а з іншого – одним з прийнятних рішень, яке може привести до формування цільового S-блоку. Аналіз отриманих результатів вказує на те, що середнє арифметичне значення нелінійності $N_f$ досягається приблизно при ітерації зовнішнього циклу, що відповідає поточної температурі знайденого мінімуму ($T_0 = 20\,000 \ldots 40\,000$).

Більш висока температура приводить до появлення так званих *непродуктивних погіршень*, тобто погіршення, які приводять до постійного відкату найденого рішення до погіршеного стану. Тому більшу кількість ітерацій, яка здійснена при непродуктивних погіршеннях, можливо також віднести до *непродуктивних ітерацій*, тобто таких, які не приводять до покращення загального стану системи.

З аналізу результатів $N_f$ видно, що у правій частині середні арифметичні значення $N_f$ досягають значень, які відповідають лівій частині після першої ітерації зовнішнього циклу, лише приблизно після кількості ітерацій, які приводять поточну температуру до значень знайденого мінімуму ($T_0 = 20\,000 \ldots 40\,000$).

Також змінюється час пошуку цільового S-блоку. Починаючи з малих значень $T_0$, при поступовому збільшені, час пошуку скорочується та в кінці середнього етапу може складати у 1,5 – 2 рази менше значення. Потім, враховуючи значну кількість непродуктивних погіршень, час пошуку значно зростає. Чим вище значення $T_0$, тім більше така кількість непродуктивних погіршень, чим вище значення $\alpha$, тим довше воно триває.

При високому значенні початкової температури або низькій швидкості її знижування, необхідна значна кількість зовнішніх циклів для стабілізації системи у деякому локальному мінімумі. При недостатній максимальній кількості зовнішніх циклів алгоритм може не знайти локальний мінімум, що приведе до значного зниження кількості знайдених рішень або їх повної відсутності.

Значення початкової температури, при якій ймовірність знайти цільовий S-блок є максимальною та ще не спостерігаються непродуктивні ітерації, будемо називати *оптимальною температурою* (позначимо як $T_0^{\text{opt}}$). Знайдений мінімум середнього часу формування цільового S-блоку відповідає інтервалу початкової температури $T_0^{\text{opt}} = 20\,000 - 40\,000$. Зі збільшенням параметру $\alpha$ мінімум зсувається в бік меншого значення $T_0$.

Для підвищення точності отриманих значень було підвищено кількість запусків для кожної температури до 1000. Для прийнятного часу тестування було зменшено діапазон значень $T_0 = 12500 - 37500$ та протестовано всього 11 значень $T_0$ при трьох значеннях $\alpha = 0{,}85; 0{,}9; 0{,}95$ (тестування проведено за 77 годин). Результати ймовірності формування цільового S-блоку та середній час формування наведено на рис. 20 та 21, пунктир – апроксимаційне значення.

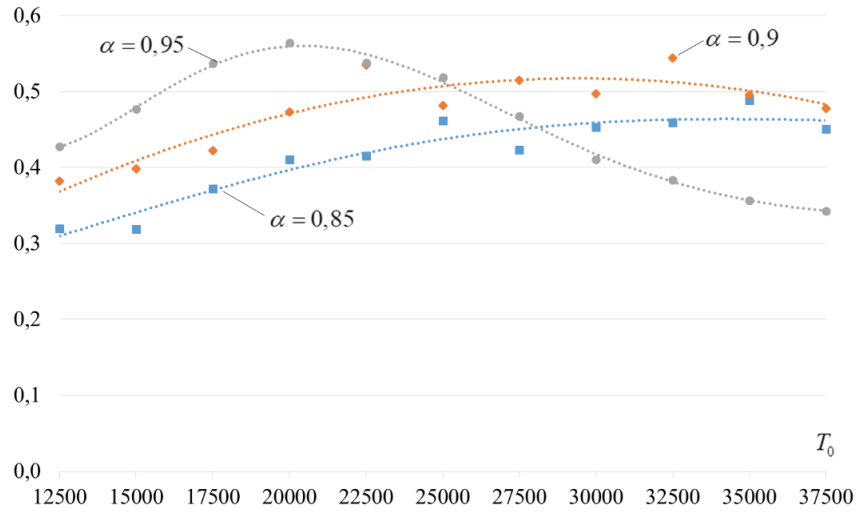


Рис. 20. Ймовірність формування цільового S-блоку при $\alpha = 0{,}85;\ 0{,}9;\ 0{,}95$ та max_inner_loops=650
(кожна точка – середнє значення за 1000 тестів)

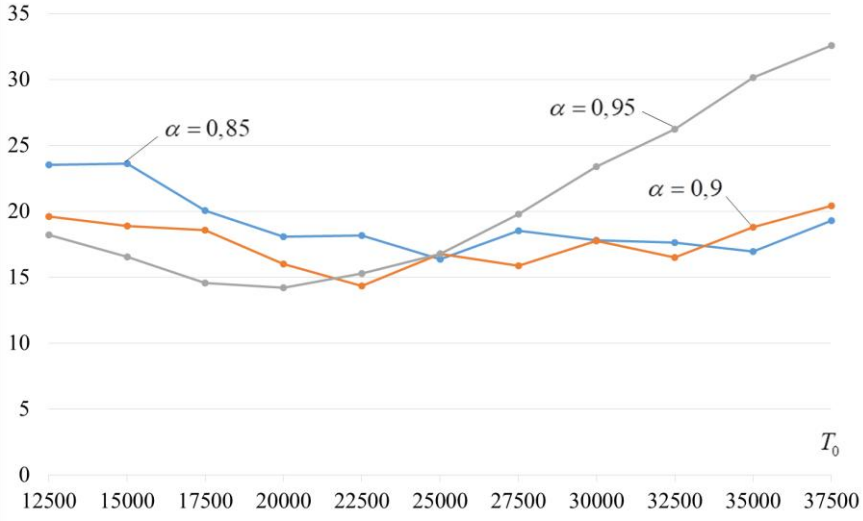


Рис. 21. Середній час (с) формування цільового S-блоку при $\alpha = 0{,}85;\ 0{,}9;\ 0{,}95$ та max_inner_loops=650
(кожна точка – середнє значення за 1000 тестів)

Згідно з наведеними даними при обраних параметрах (max_outer_loops=50, max_inner_loops=650, max_frozen_outer_loops=5, thread_count=30), найкращі результати отримано при $\alpha = 0{,}95$ та $T_0 = 20\,000$. Ймовірність знайти цільовий S-блок (з $N_f = 104$) дорівнює 56,4 %, а середній час пошуку складає 14,2 с.

Для порівняння отриманих результатів із іншими відомими реалізаціями SA алгоритму в табл. 2 наведено оцінки складності пошуку цільового S-блоку (із нелінійністю $N_f$).

Таблиця 2

Порівняння отриманих результатів з генерації бієктивних 8-бітних S-блоків
(для різних реалізацій SA алгоритму)

| Параметр | [8, 22] | [20] | [19] | **Our work** |
|---|---|---|---|---|
| Найвище значення $N_f$, що було отримано у знайденому S-блоку | 102 | 92 | 104 | **104** |
| Ймовірність формування S-блоку | 1/200 = 0,5% | – | – | **56,4 %** |
| Час генерації (пошуку) S-блоку | – | – | – | **14,2 с** |
| Складність генерації (кількість ітерацій пошуку) | – | – | 30,000,000 | **450,000** |

Позначеннями «–» в табл. 2 наведено випадки з невизначеними показниками.

**Висновки**

За результатами досліджень робимо висновок, що метод імітації відпалу добре проводить пошук цільового (тобто з заданими властивостями) S-блоку. При вдало підібраних параметрах алгоритму ймовірність знайти S-блок з нелінійністю $N_f = 104$ буде дорівнювати майже одиниці. Однак 100 % ймовірність знайти цільовий S-блок не є оптимальним шляхом з точки зору затраченого на пошук часу. Введення додаткових обмежень зменшує час, що затрачується при кожній спробі, але й зменшує ймовірність знайти цільовий S-блок у кожній спробі. Тому результати пошуку за допомогою методу імітації відпалу дуже чутливі до майже всіх вхідних параметрів пошуку, а їх оптимізація є дуже трудомістким процесом.

Під час дослідження було вивчено вплив вхідних параметрів методу імітації відпалу на результат пошуку цільового S-блоку. За результатами досліджень наведено порівняльні характеристики часу пошуку та внутрішніх станів алгоритму, проведено оптимізацію за критерієм мінімізації часу пошуку.

При обраних параметрах алгоритму (max_outer_loops=50, max_inner_loops=650, max_frozen_outer_loops=5, thread_count=30) найкращі результати було отримано при $\alpha = 0,95$ та $T_0 = 20\,000$. При цьому ймовірність знайти цільовий S-блок (з $N_f = 104$) дорівнює 56,4 %, а середній час пошуку складає 14,2 с. При цьому алгоритм потребує в середньому близько 450,000 ітерацій пошуку. При підвищенні кількості внутрішніх ітерацій ймовірність знайти цільовий S-блок підвищується до 97 %. Це кращий із відомих результатів застосування SA алгоритму для генерації бієктивних 8-бітних S-блоків.

**Список літератури:**


1. Álvarez-Cubero J. Vector Boolean Functions: applications in symmetric cryptography, (2015). https://doi.org/10.13140/RG.2.2.12540.23685.
2. Cusick T., Stănică P. Cryptographic Boolean Functions and Applications: Second edition. (2017).
3. Freyre Echevarría A. Evolución híbrida de s-cajas no lineales resistentes a ataques de potencia, (2020). https://doi.org/10.13140/RG.2.2.17037.77284/1.
4. Schneier B. Applied cryptography: protocols, algorithms, and source code in C. New York : Wiley (1996).
5. Menezes A.J., Oorschot P.C. van, Vanstone S.A., Oorschot P.C. van, Vanstone S.A. Handbook of Applied Cryptography. CRC Press (2018). https://doi.org/10.1201/9780429466335.
6. Kuznetsov A.A., Potii O.V., Poluyanenko N.A., Gorbenko Y.I., Kryvinska N. Stream Ciphers in Modern Real-time IT Systems: Analysis, Design and Comparative Studies. Springer International Publishing (2022). https://doi.org/10.1007/978-3-030-79770-6.
7. Delahaye D., Chaimatanan S., Mongeau M. Simulated annealing: From basics to applications. In: Gendreau, M. and Potvin, J.-Y. (eds.) Handbook of Metaheuristics. pp. 1-35.ISBN 978-3-319-91085–7. Springer (2019). https://doi.org/10.1007/978-3-319-91086-4_1.
8. Clark J.A., Jacob J.L., Stepney S. The design of s-boxes by simulated annealing. In: Proceedings of the 2004 Congress on Evolutionary Computation (IEEE Cat. No.04TH8753). pp. 1533-1537 Vol.2 (2004). https://doi.org/10.1109/CEC.2004.1331078.
9. Tesar P. A New Method for Generating High Non-linearity S-Boxes. (2010).
10. Picek S., Cupic M., Rotim L. A New Cost Function for Evolution of S-Boxes. Evolutionary Computation. 24, 695–718 (2016). https://doi.org/10.1162/EVCO_a_00191.
11. Freyre-Echevarría A., Alanezi A., Martínez-Díaz I., Ahmad M., Abd El-Latif A.A., Kolivand, H., Razaq, A. An External Parameter Independent Novel Cost Function for Evolving Bijective Substitution-Boxes. Symmetry. 12, 1896 (2020). https://doi.org/10.3390/sym12111896.
12. Freyre Echevarría A., Martínez Díaz I. A new cost function to improve nonlinearity of bijective S-boxes. (2020).
13. Kuznetsov A., Poluyanenko N., Kandii S., Zaichenko Y., Prokopovich-Tkachenko D., Katkova T. WHS Cost Function for Generating S-boxes // 2021 IEEE 8th International Conference on Problems of Infocommunications, Science and Technology (PIC S T). pp. 434–438 (2021). https://doi.org/10.1109/PICST54195.2021.9772133.
14. Ivanov G., Nikolov N., Nikova S. Reversed genetic algorithms for generation of bijective s-boxes with good cryptographic properties. Cryptogr. Commun. 8, 247–276 (2016). https://doi.org/10.1007/s12095-015-0170-5.
15. Rodinko M., Oliynykov R., Gorbenko Y. Optimization of the High Nonlinear S-Boxes Generation Method. Tatra Mountains Mathematical Publications. 70, 93–105 (2017). https://doi.org/10.1515/tmmp-2017-0020.
16. Kuznetsov A., Poluyanenko N., Kandii S., Zaichenko Y., Prokopovich-Tkachenko D., Katkova T. Optimizing the Local Search Algorithm for Generating S-Boxes // 2021 IEEE 8th International Conference on Problems of

Infocommunications, Science and Technology (PIC S T). pp. 458–464 (2021). https://doi.org/10.1109/PICST54195.2021.9772163.

17. Chen G. A novel heuristic method for obtaining S-boxes. Chaos, Solitons & Fractals. 36, 1028–1036 (2008). https://doi.org/10.1016/j.chaos.2006.08.003.

18. Souravlias D., Parsopoulos K.E., Meletiou G.C. Designing bijective S-boxes using Algorithm Portfolios with limited time budgets. Applied Soft Computing. 59, 475–486 (2017). https://doi.org/10.1016/j.asoc.2017.05.052.

19. McLaughlin J., Clark J.A. Using evolutionary computation to create vectorial Boolean functions with low differential uniformity and high nonlinearity. arXiv (2013). https://doi.org/10.48550/arXiv.1301.6972.

20. Wang J., Zhu Y., Zhou C., Qi Z. Construction Method and Performance Analysis of Chaotic S-Box Based on a Memorable Simulated Annealing Algorithm. Symmetry. 12, 2115 (2020). https://doi.org/10.3390/sym12122115.

21. McLaughlin J. Applications of search techniques to cryptanalysis and the construction of cipher components, https://etheses.whiterose.ac.uk/3674/, (2012).

22. Ivanov G., Nikolov N., Nikova S. Cryptographically Strong S-Boxes Generated by Modified Immune Algorithm // Pasalic, E. and Knudsen, L.R. (eds.) Cryptography and Information Security in the Balkans. pp. 31–42. Springer International Publishing, Cham (2016). https://doi.org/10.1007/978-3-319-29172-7_3.

*Надійшла до редколегії 10.05.2022*

*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

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

**Логачова Єлизавета Олегівна –** студентка кафедри безпеки інформаційних систем і технологій, факультет комп'ютерних наук; Харківський національний університет імені В.Н. Каразіна, Україна; e-mail: lohachova2020kb11@student.karazin.ua, ORCID: https:// orcid.org/0000-0002-9815-466X