ISSN 2519-2310

CS&CS Journal

KARAZIN UNIVERSITY
CLASSICS AHEAD OF TIME

2(14) 2019

COMPUTER SCIENCE AND CYBERSECURITY

КОМП'ЮТЕРНІ НАУКИ ТА КІБЕРБЕЗПЕКА

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ УКРАИНЫ
MINISTRY OF EDUCATION AND SCIENCE OF UKRAINE

ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ імені В.Н.КАРАЗІНА
ХАРЬКОВСКИЙ НАЦИОНАЛЬНЫЙ УНИВЕРСИТЕТ имени В.Н. КАРАЗИНА
V.N. KARAZIN KHARKIV NATIONAL UNIVERSITY

КОМП'ЮТЕРНІ НАУКИ ТА КІБЕРБЕЗПЕКА
КОМПЬЮТЕРНЫЕ НАУКИ И КИБЕРБЕЗОПАСНОСТЬ
COMPUTER SCIENCE AND CYBERSECURITY
(CS&CS)

Issue 2(14) 2019

Заснований 2015 року

Міжнародний електронний науково-теоретичний журнал
Международный электронный научно-теоретический журнал
International electronic scientific journal

The journal publishes research articles on theoretical, scientific and technical problems of effective facilities development for computer information communication systems and on information security problems based on advanced mathematical methods, information technologies and technical means.

Journal is published quarterly.
Approved for placement on the Internet by Academic Council of the Karazin Kharkiv National University (December 23, 2019, protocol No. 13)

The journal has Digital Object Identifier**: 10.26565/2519-2310**.

Editorial office:
V.N. Karazin Kharkiv National University
Svobody Sq., 6, office 315a, Kharkiv, 61022, Ukraine *(North building of University, 3th floor)*
Phone: +38 (057) 705-10-83
E-mail: *cscsjournal@karazin.ua*
Web-page: *http://periodicals.karazin.ua/cscs (Open Journal System)*

Published articles have been internally and externally peer reviewed

TABLE OF CONTENTS

Issue 2(14) 2019

УДК 681: 519 DOI: 10.26565/2519-2310-2019-2-01

МАТЕМАТИЧНА МОДЕЛЬ ПРОЦЕСУ ТАБЛИЧНОЇ РЕАЛІЗАЦІЇ ОПЕРАЦІЇ АЛГЕБРАЇЧНОГО МНОЖЕННЯ В КЛАСІ ЗАЛИШКІВ

Віктор Краснобаєв, Михайло Зуб, Олеся Решетняк,
Андрій Д'яченко, Ірина Локоткова, Костянтин Мисливцев

Харківський національний університет імені В.Н. Каразіна, майдан Свободи, 4, Харків, 61022, Україна
v.a.krasnobaev@gmail.com, mishazub007@gmail.com, lesyandr13@gmail.com
andrey.090220@gmail.com, lokotosyk@ukr.net, kostyavtx@gmail.com

Рецензент: Олександр Кузнецов, д.т.н., проф., академік Академії наук прикладної радіоелектроніки, Харківський національний університет імені В.Н. Каразіна, майдан Свободи 6, Харків, 61022, Україна
kuznetsov@karazin.ua

Надійшло: Вересень 2019.

Анотація: *На підставі властивостей класу залишків в статті синтезована математична модель процесу табличної реалізації модульного множення, як для позитивного, так і для негативного числових діапазонів обробки інформації засобами обробки цілочисельної інформації. Дана модель рекомендована до практичного застосування при розробці методів і алгоритмів швидкої обробки криптографічної інформації. Пошук шляхів спрощення структури табличного операційного пристрою засобу обробки цілочисельної інформації зумовив необхідність вдосконалення математичної моделі, методів і алгоритмів реалізації модульних операцій, що дозволяють підвищити ефективність застосування табличній арифметики в класі залишків. Особливістю реалізації даної моделі є можливість зменшення кількості обладнання операційного пристрою засобу обробки цілочисельної інформації за рахунок скорочення на (50-70)% логічних елементів "І" в вузлах таблиці постійний запам'ятовуючий пристрій, безпосередньо реалізують операцію модульного множення за довільним m_i модулю класу залишків. Це можливо за рахунок використання властивостей симетрії таблиці реалізації $a_i b_i (\mathrm{mod}\ m_i)$ модульної операції множення.*

Ключові слова: *клас залишків; реалізація таблиць; математична модель; операція алгебраїчне множення.*

1 Вступ

Складність завдань, які вирішуються сучасними системами і засобами обробки цілочисельної інформації (ЗОІ), обумовлює до них підвищені вимоги щодо якості обробки інформації [1-8]. Рішення широкого кола складних обчислювальних задач вимагає (наприклад, реалізація асиметричних криптографічних алгоритмів, оптимізація обчислень в теорії сигналів та шумостійке кодування [4, 8-11], реалізація симетричних крипто-примітивів [12-18] та багато іншого [19-22]) значних обсягів розрахунків, які проводяться в реальному часі функціонування ЗОІ. В цьому аспекті пошук методів і засобів підвищення продуктивності обробки цифрової інформації, є актуальним [23-30].

У будь-якій позиційній системі числення (ПСЧ) виконання арифметичних операцій передбачає послідовну обробку всіх розрядів операндів за правилами, що визначаються змістом даної операції, і не може бути закінчено до тих пір, поки не будуть послідовно визначені значення всіх проміжних результатів з урахуванням всіх міжрозрядних зв'язків операндів. Таким чином, ПСЧ, в якій представляється і обробляється інформація в сучасних обчислювальних машинах, мають істотний недолік - наявність логічних зв'язків між двійковими розрядами в оброблюваних операндах. Даний недолік робить істотний вплив на методи реалізації арифметичних операцій, ускладнюють апаратуру і обмежують швидкодію. Тому природно вишукування такої машинної арифметики, в якій би порозрядні зв'язку були б ослаблені, або відсутні взагалі. Відзначимо, будь-яка система числення в більшій мірі впливає на структуру і принципи функціонування операційного пристрою (ОП) ЗОІ.

Пошук шляхів підвищення продуктивності обробки інформації позиційних ЗОІ реального часу привів до необхідності проведення досліджень можливості використання табличного методу (табличної арифметики) реалізації модульних операцій. У загально-

му випадку табличне ОП ЗОІ, призначене для реалізації арифметичних операцій (які реалізується в унітарній коді), являє собою двовхідний постійний запам'ятовуючий пристрій (ПЗП). Для кожного з входів ПЗП кількість вхідних шин для *l*-байтового (8l двійкових розряди) ОП одно 28l. При цьому загальна кількість логічних схем збігу "І" в вузлах ПЗП (яке в основному і визначає загальну кількість обладнання табличного ОП ЗОІ) рівно

$$N_{l\,ПСЧ} = 2^{8l} \times 2^{8l} = 2^{16l}.$$

Виходячи з цього, очевидно, що таблична реалізація цілочисельних модульних арифметичних операцій в ПСЧ може бути доцільна тільки для значення *l=1*. Дійсно, в цьому випадку

$$N_{1ПСЧ} = 2^{16} = 65536,$$

що є прийнятним кількістю обладнання ОП для сучасного розвитку елементної бази. Однак, як зазначено вище, тенденція розвитку систем і засобів обробки криптографічного інформації (ЗОКІ) спрямована на збільшення довжини розрядної сітки ЗОІ. Вже зараз необхідно для практичного використання ЗОКІ з l = 4 і l = 8. В цьому випадку

$$N_{4ПСЧ}=2^{32}\times 2^{32}=2^{64} \text{ і } N_{8ПСЧ}=2^{64}\times 2^{64}=2^{128}.$$

Якщо врахувати, наприклад, що 2^{32}=4294967296, 2^{64}=18446744073709551616, а $2^{128} \approx 3{,}4\times 10^{38}$, то очевидно, що табличний метод реалізації арифметичних операцій в ПСЧ практично не можливо застосувати.

Інші результати використання табличного методу реалізації арифметичних операцій можна отримати, якщо розглянути ЗОКІ в непозиційній системі числення класу залишків (КЗ). Клас залишків має унікальну властивість незалежності один від одного залишків за прийнятою системі підстав. Ця незалежність відкриває широкі можливості у побудові не тільки нової машинної арифметики, а й принципово нової схемної реалізації ЗОКІ, яка в свою чергу помітно розширює застосування табличної арифметики [1-3, 23-26]. Дійсно, в загальному випадку, для *l*-байтових машинних слів, при реалізації алгоритмів модульної обробки інформації для табличного ОП ЗОІ в КЗ не обходимо

$$N_{lКЗ} = \sum_{i=1}^{n} m_i^2$$

схем збігу І, а для ОП ЗОІ в КЗ з l=4 и l=8 відповідно маємо $N_{4КЗ}$=2397 і $N_{8КЗ}$=13275, що цілком прийнято при реалізації арифметичних операцій додавання, віднімання і множення тим більше використовуючи сучасну елементну мікроелектронну базу (НВІС, ПЛМ або ПЛІС). Вищезазначена обставина підтверджує доцільність і ефективність проведення практичних досліджень та розробки табличних методів і алгоритмів реалізації модульних операцій в КЗ.

Відзначимо основні переваги табличного методу побудови ОП ЗОІ в КЗ. По-перше, високу швидкодію виконання арифметичних операцій. Результат операції може бути отриманий в момент надходження вхідних операндів в табличний процесор, тобто практично за один такт роботи ЗОІ. Таким чином, час виконання арифметичних операцій в КЗ порівняно з тактовою частотою обчислювача, що принципово неможливо для позиційних обчислювальних машин при існуючій елементній базі. По-друге, табличні ОП мають високу надійність, так як реалізуються у вигляді набору, по числу n модулів m_i КЗ, компактних ПЗП. В цьому випадку весь тракт, що складається з *n* (по числу модулів m_i КЗ), обробки інформації ОП будується за блоковим принципом, що покращує безвідмовність і ремонтопридатність ЗОІ. По-третє, відзначимо простоту технічної реалізації ОП ЗОІ в КЗ, що складається в основному з регістрів, табличних суматорів, шифраторів і дешифраторів і уніфікацію його обладнання для довільного модуля $\{m_i\}, i=\overline{1,n}.$

Відзначимо, необхідність розробки методів і алгоритмів арифметичного множення чисел необхідно попередньо мати і надалі використовувати математичну модель (ММ) процесу табличній реалізації операції модульного множення в КЗ.

2 Огляд джерел

У літературі описані ММ, на підставі яких реалізовані табличні методи і алгоритми модульного множення чисел в непозиційної системі числення КЗ [1-3, 27-30]. Пошук шляхів спрощення структури табличного ОП ЗОІ зумовив необхідність вдосконалення ММ, методів і алгоритмів реалізації модульних операцій, що дозволяють підвищити ефективність застосування табличній арифметики в КЗ. Так в [1, 2, 23] представлена ММ процесу табличної реалізації операції арифметичного модульного множення в КЗ. Особливістю реалізації даної моделі є можливість зменшення кількості обладнання ОП ЗОІ за рахунок скорочення на (50-70)% логічних елементів "І" в вузлах таблиці ПЗП, безпосередньо реалізують операцію модульного множення за довільним m_i модулю КЗ. Це можливо за рахунок використання властивостей симетрії таблиці реалізації $a_i b_i (\mathrm{mod}\, m_i)$ модульної операції множення.

В КЗ число А представляється у вигляді сукупності залишків $\{a_i\}$ по n модулях (підстав) $\{m_i\}$, де:

$$a_i = A - [A / m_i] \cdot m_i; \quad M = \prod_{i=1}^{n} m_i .$$

У цьому випадку число А в КЗ представляється в наступному вигляді

$$A = (a_1, a_2, ..., a_n).$$

Нехай задана пара операндів

$$A = (a_1, ..., a_n) \text{ та } B = (b_1, ..., b_n)$$

в КЗ з попарно взаємно простими основами $m_1,\ldots,m_n$.. Відповідно до правил виконання арифметичних операцій в КЗ кожній парі залишків a_i і b_i ставлять у відповідність величина $(a_i \otimes b_i) \bmod m_i$. Таким чином, весь машинний тракт обчислювальної операції $(A \otimes B) \bmod M$ в КЗ можна представити у вигляді n незалежних однотипних ПЗП.

Розглянемо процедуру реалізації операції модульного множення (найбільш трудомістку арифметичну операцію). Складемо таблицю з числових значень $a_i b_i (\mathrm{mod}\, m_i)$. Ця таблиця симетрична щодо діагоналей, вертикалі і горизонталі, що проходять між числами

$$\frac{(m_i - 1)}{2} \text{ та } \frac{(m_i + 1)}{2} \text{ для } m_i \text{ - непарного числа.}$$

Симетричність щодо лівої діагоналі визначається комутативністю операції множення

$$a_i \cdot b_i = b_i \cdot a_i .$$

Симетричність щодо правої діагоналі визначається тим, що

$$a_i \cdot b_i \equiv [(m_i - b_i)(m_i - a_i)] \bmod m_i .$$

Симетричність щодо вертикалі і горизонталі визначається з умови кратності по модулю m_i суми симетричних чисел:

$$a_i b_i \equiv [m_i - a_i (m_i - b_i)] \bmod m_i ,$$

$$a_i b_i \equiv [m_i - b_i (m_i - a_i)] \bmod m_i .$$

Використовуючи властивості симетрії можна повністю відновити таблицю даних модульного множення $a_i b_i (\mathrm{mod}\, m_i)$. Для цього достатньо мати числову інформацію тільки її восьмий частини. Звідси виникає можливість спростити таблицю (зменшити кількість схем збігу ПЗП) модульного множення.

Для реалізації операції $a_i b_i (\mathrm{mod}\, m_i)$ має бути ефективним, з точки зору швидкодії виконання операції множення, в чотири рази зменшити таблицю модульного множення. Для вирішення поставленого завдання необхідно ввести ознака, що визначає місце розташування вхідних чисел в кожному з чотирьох квадрантів таблиці $a_i b_i (\mathrm{mod}\, m_i)$.

Розглянемо один з можливих варіантів кодування вхідних операндів a_i і b_i таблиці по модулю m_i за допомогою коду інформаційного стиснення даних (КІСД).

Нехай дано вхідні операнди a_i та b_i. Значення a_i (або b_i), що лежать в діапазоні $\left[0, \frac{m_i-1}{2}\right)$, можуть бути закодовані довільним чином, а значення a_i (або b_i), що лежать в діапазоні $\left[\frac{m_1+1}{2}, m_i-1\right)$, кодуються, як $m_i - a_i$(або $m_i - b_i$). Для відмінності діапазонів вводиться ознака γ_a (γ_b) КІСД, певний в такий спосіб:

$$\gamma_a\ (\gamma_b) = \begin{cases} 0, \textit{ якщо } 0 \le a_i(b_i) \le \dfrac{m_i-1}{2}, \\ 1, \textit{ якщо } \dfrac{m_i+1}{2} \le a_i(b_i) \le m_i - 1, \end{cases}$$

де $0 \le a^*(b^*) \le (m_i - 1)/2.$

Процедура визначення результату операції модульного множення, за допомогою введеного коду інформаційного стиснення даних, наступний: якщо задані два числа в КІСД виду

$$a_i = (\gamma_a, a_i^*), b_i = (\gamma_b, b_i^*),$$

то для отримання твіру цих чисел по модулю m_i, досить отримати добуток $a_i^* b_i^* (\mathrm{mod}\, m_i)$ і інвертувати його узагальнену ознаку γ_i в разі, якщо γ_a відмінно від γ_b, тобто

$$a_i b_i (\mathrm{mod}\, m_i) = (\gamma_i, a_i^* b_i^* (\mathrm{mod}\, m_i)), \quad \text{де } \gamma_i = \begin{cases} \overline{\gamma_i}, \textit{ якщо } \gamma_a \ne \gamma_b, \\ \gamma, \textit{ якщо } \gamma_a = \gamma_b. \end{cases}$$

Запропонований варіант реалізації модульних операцій в КЗ дозволяють оптимізувати структуру ЗОІ шляхом підвищення ефективності використання табличній арифметики. Скорочення кількості обладнання ПЗП, що складають основну частину ОП, дозволяє підвищити на дійсні показники (збільшити ймовірність безвідмовної роботи P(t), зменшити час відновлення $T_в$) і поліпшити експлуатаційно-технічні показники (зменшити масу і габаритні розміри, зменшити споживану потужність і поліпшити технічне обслуговування ЗОІ в КЗ).

З огляду на КІСД, математична модель процесу табличної реалізації в позитивному числовому діапазоні двох чисел в КЗ випаде наступними математичними співвідношеннями:

$$\begin{aligned} C = A \cdot B (\mathrm{mod}\, M) = [(a_1, a_2, ..., a_i, ..., a_n) \cdot (b_1, b_2, ..., \\ ,..., b_i, ..., b_n)] \mathrm{mod}\, M = [(a_1 \cdot b_1) \mathrm{mod}\, m_1, (a_2 \cdot b_2) \mathrm{mod}\, m_2, ..., \\ ,..., (a_i \cdot b_i) \mathrm{mod}\, m_i, ..., (a_n \cdot b_n) \mathrm{mod}\, m_n] = \\ = \{[(\gamma_{a_1}, a_1^*) \cdot (\gamma_{b_1}, b_1^*)] \mathrm{mod}\, m_1, [(\gamma_{a_2}, a_2^*) \cdot \\ \cdot (\gamma_{b_2}, b_2^*) \mathrm{mod}\, m_2], ..., (\gamma_{a_i}, a_i^*) \cdot (\gamma_{b_i}, b_i^*) \mathrm{mod}\, m_i], ..., \\ ,..., [(\gamma_{a_n}, a_n^*) \cdot (\gamma_{b_n}, b_n^*)] \mathrm{mod}\, m_n\} = \{[\gamma_1, (a_1^* \cdot b_1^*) \mathrm{mod}\, m_1], [\gamma_2, (a_2^* \cdot b_2^*) \mathrm{mod}\, m_2], ..., \\ ,..., [\gamma_i, (a_i^* \cdot b_i^*) \mathrm{mod}\, m_i], ..., [\gamma_n, (a_n^* \cdot b_n^*) \mathrm{mod}\, m_n]\} = (c_1, c_2, ..., c_i, ..., c_n). \end{aligned} \tag{1}$$

При цьому ознака γ_{a_i} (γ_{b_i}) коду (γ_{a_i}, a_i^*) $((\gamma_{b_i}, b_i^*))$ КІСД таблиці модульного множення для довільного m_i модуля КЗ визначається наступним чином. Для m_i - парного

$$\gamma_{a_i}(\gamma_{b_i}) = \begin{cases} 0, \textit{ якщо } 0 \le a_i(b_i) \le m_i/2, \\ 1, \textit{ якщо } m_i/2 < a_i(b_i) \le m_i - 1. \end{cases} \tag{2}$$

Для m_i - непарного

$$\gamma_{a_i}(\gamma_{b_i}) = \begin{cases} 0, & якщо\ 0 \le a_i(b_i) \le (m_i - 1)/2, \\ 1, & якщо\ (m_i - 1)/2 < a_i\ (b_i) \le m_i - 1. \end{cases} \quad (3)$$

при цьому $0 \le a_i(b_i) \le m_i - 1$.

Числова частина $a_i^*\ (b_i^*)$ КІСД визначається так. Для m_i - парного це буде

$$a_i^*\ (b_i^*) = \begin{cases} a_i\ (b_i), якщо\ 0 \le a_i(b_i) \le m_i/2; \\ \overline{a_i}\ (\overline{b_i}) = m_i - a_i(b_i), якщо\ m_i/2 < a_i(b_i) \le m_i - 1, \end{cases} \quad (4)$$

при цьому $0 \le a_i^*(b_i^*) \le m_i/2$.

Для m_i -непарного

$$a_i^*(b_i^*) = \begin{cases} a_i(b_i), якщо\ 0 \le a_i(b_i) \le (m_i - 1)/2, \\ \overline{a_i}(\overline{b_i}) = m_i - a_i(b_i), якщо\ (m_i - 1)/2 < a_i(b_i) \le m_i - 1, \end{cases} \quad (5)$$

при цьому $0 \le a_i^*(b_i^*) \le (m_i - 1)/2$.

Якщо $(a_i \cdot b_i) \bmod m_i$ модульного множення визначається в КІСД як $[\gamma_i, (a_i^* \cdot b_i^*) \bmod m_i]$, тоді

$$(a_i \cdot b_i) \bmod m_i = \begin{cases} (a_i^* \cdot b_i^*) \bmod m_i, \\ якщо\ (\gamma_{a_i} + \gamma_{b_i}) = 0 \pmod 2; \\ m_i - (a_i^* \cdot b_i^*) \bmod m_i, \\ якщо\ (\gamma_{a_i} + \gamma_{b_i}) = 1 \pmod 2, \end{cases} \quad (6)$$

при цьому $0 \le (a_i^* \cdot b_i^*) \bmod m_i \le m_i - 1$.

Таким чином, сукупність виразів (2) - (6) є ММ процесу табличної реалізації модульного арифметичного множення в КЗ.

Недолік розглянутої ММ полягає в тому, що її використання не дає можливості створити табличний метод реалізації операції алгебраїчного множення в КЗ.

Мета статті - розробити ММ процесу табличної реалізації множення в КЗ як для позитивного, так і для негативного числових діапазонів.

2 Основна частина

Для побудови ММ процесу табличної реалізації множення в КЗ, як для позитивного, так і для негативного числових діапазонів представимо вхідні числа А і В у наступному вигляді (штучна форма представлення чисел в КЗ [4])

$A' = A + m/2$ та $B' = B + m/2$, для m – парних чисел;

$A' = A + (m-1)/2$ та $B' = B + (m-1)/2$, для m – непарних чисел.

Якщо, наприклад, m парне число, тоді виконуються наступні співвідношення

$$\begin{cases} -m/2 \le A(B) < m/2, \\ 0 \le A'(B') < m - 1, \\ -m/2 \le A \cdot B < m/2, \\ 0 \le (A \cdot B)' < m - 1. \end{cases}$$

Очевидно, що

$$(A \cdot B)' = A \cdot B + m/2. \quad (7)$$

Тоді маємо

$$(A'\cdot B')\bmod m=\left[\ (A+\ {}^{m}\!/_{2})(B+\ {}^{m}\!/_{2})\ \right]\bmod m=$$
$$=\left[\ AB(\bmod\ {}^{m}\!/_{2})+{}^{m}\!/_{2}\cdot(A+B+\ {}^{m}\!/_{2})\ \right]\bmod m. \quad (8)$$

З виразу (8) очевидно, що

$$A\cdot B=A'\cdot B'-{}^{m}\!/_{2}\cdot(A+B+{}^{m}\!/_{2}) \quad (9)$$

Підставимо вираз (9) в формулу (7). Отримаємо, що

$$(A\cdot B)'=A'\cdot B'-{}^{m}\!/_{2}\cdot(A+B+{}^{m}\!/_{2})+{}^{m}\!/_{2} \quad (10)$$

У виразі (10) є член, який має числове значення m / 2. Він і обумовлює помилку в обчисленні значення $A'\cdot B'(\bmod m)$. Таким чином формули для обчислення AB (mod m) мають такий вигляд для m парних чисел

$$\left[(A\cdot B)\bmod\ {}^{m}\!/_{2}\ \right]'=(A'\cdot B')\bmod m+{}^{m}\!/_{2}, \quad (11)$$

або

$$\left[\ (A\cdot B)\bmod\ {}^{m}\!/_{2}\ \right]'=(A'\cdot B')\bmod m. \quad (12)$$

Для m непарного маємо

$$\left[\ (A\cdot B)\bmod{}^{(m-1)}\!/_{2}\ \right]'=(A'\cdot B')\bmod m+{}^{(m-1)}\!/_{2}, \quad (13)$$

або

$$\left[\ (A\cdot B)\bmod{}^{(m-1)}\!/_{2}\ \right]'=(A'\cdot B')\bmod m. \quad (14)$$

З огляду на вираження (7) ÷ (14), побудуємо ММ процесу модульного множення для позитивного і негативного (алгебраїчне множення) цілочисельних числових діапазонів.

$$a_i'=a_i+m_i/2,\ a_i'=a_i'+(m_i-1)/2;\ a_i'=[\gamma_{a_i}',(a_i')^*]$$
$$b_i'=b_i+m_i/2,\ b_i'=b_i'+(m_i-1)/2;\ b_i'=[\gamma_{b_i}',(b_i')^*] \quad (15)$$

Для m_i - парного

$$\gamma_{a_i}'(\gamma_{b_i}')=\begin{cases}0,\ якщо\ 0\le a_i'(b_i')\le m_i/2,\\ 1,\ якщо\ m_i/2<a_i'(b_i')\le m_i-1.\end{cases} \quad (16)$$

Для m_i - непарного

$$\gamma_{a_i}'(\gamma_{b_i}')=\begin{cases}0,\ якщо\ 0\le a_i'(b_i')\le (m_i-1)/2,\\ 1,\ якщо\ (m_i-1)/2<a_i'(b_i')\le m_i-1.\end{cases} \quad (17)$$

Числова частина $(a_i')^*\ [(b_i')^*]$ КІСД визначається наступним чином. Для m_i- парного

$$(a_i')^*[(b_i')^*]=\begin{cases}a_i'\ (b_i'),\ якщо\ 0\le a_i'(b_i')\le m_i/2;\\ \overline{a_i'}\ (\overline{b_i'})=m_i-a_i'(b_i'),\\ якщо\ m_i/2<a_i'(b_i')\le m_i-1,\end{cases} \quad (18)$$

при цьому $0\le(a_i')^*\ [(b_i')^*]\le m_i/2$. Для m_i- непарного числа

$$(a_i')^*[(b_i')^*]=\begin{cases}a_i'\ (b_i'),\ якщо\ 0\le a_i'(b_i')\le (m_i-1)/2;\\ \overline{a_i'}\ (\overline{b_i'})=m_i-a_i'(b_i'),\\ якщо\ (m_i-1)/2<a_i'(b_i')\le m_i-1,\end{cases} \quad (19)$$

при цьому $0 \le (a_i')^* [(b_i')^*] \le (m_i - 1)/2$.

Результат $(a_i' \cdot b_i') \bmod m_i$ операції визначається в КІСД, тобто у вигляді

$$\left\{\gamma_i', [(a_i')^* (b_i')^*] \bmod m_i\right\},$$

тоді

$$(a_i' \cdot b_i') \bmod m_i = \begin{cases} [(a_i')^* \cdot (b_i')^*] \bmod m_i, \\ якщо\ (\gamma'_{a_i} + \gamma'_{b_i}) = 0 \pmod 2; \\ m_i - [(a_i')^* \cdot (b_i')^*] \bmod m_i, \\ якщо\ (\gamma'_{a_i} + \gamma'_{b_i}) = 1 \pmod 2, \end{cases} \tag{20}$$

при цьому $0 \le [(a_i')^* \cdot (b_i')^*] \bmod m_i \le m_i - 1$.

Формула для визначення добутку двох чисел в КЗ має такий вигляд:

$$\begin{aligned} (A \cdot B) \bmod M = (A' \cdot B') \bmod M = [(a_1', a_2', ..., a_i', ..., \\ ..., a_n') \cdot (b_1', b_2', ..., b_i', ..., b_n')] = [(a_1' \cdot b_1') \bmod m_1, \\ (a_2' \cdot b_2') \bmod m_2, ..., (a_i' \cdot b_i') \bmod m_i, ..., (a_n' \cdot b_n') \bmod m_n]. \end{aligned} \tag{21}$$

Так як усі модулі $\{m_i\}$, $i = \overline{1, n}$ КЗ, (*за винятком можливо тільки однієї основи*), непарні числа, то у подальшому, без втрати спільності міркувань, будемо вважати що основа КЗ - непарні числа. Формула (21) з урахуванням КІСД має такий вигляд

$$\begin{aligned} (A' \cdot B') \bmod M = (\{[\gamma'_{a_1}, (a_i')^*] \cdot [\gamma'_{b_1}, (b_1')^*]\} \bmod m_1, \\ \{[\gamma'_{a_2}, (a_2')^*] \times [\gamma'_{b_2}, (b_2')^*]\} \bmod m_2, ..., \{[\gamma'_{a_i}, (a_i')^*] \times \\ \times [\gamma'_{b_i}, (b_i')^*]\} \bmod m_i, ..., [\gamma'_{a_n}, (a_n')^*] \cdot [\gamma'_{b_n}, (b_n')^*]\} \bmod m_n) = \\ = (\{\gamma_1', [(a_1')^* \cdot (b_1')^*] \bmod m_1\}, \{\gamma_2', [(a_2')^* \cdot (b_2')^*] \bmod m_2\}, \\ ..., \{\gamma_i', [(a_i')^* \cdot (b_i')^*] \bmod m_i\}, ..., \{\gamma_n', [(a_n')^* \times (b_n')^*] \bmod m_n\}), \end{aligned} \tag{22}$$

де

$$\begin{aligned} (a_i' \cdot b_i') \bmod m_i = \{[\gamma'_{a_i}, (a_i')^*] \cdot [\gamma'_{b_i}, (b_i')^*]\} \bmod m_i = \\ = \{\gamma_i', [(a_i')^* \cdot (b_i')^*]\} \bmod m_i. \end{aligned} \tag{23}$$

Виходячи з (22) ÷ (23) де, а також враховуючи, що (15) ÷ (21), для m – непарного отримаємо наступні співвідношення для реалізації модульної операції алгебраїчного множення в КЗ

$$\begin{cases} \{(a_i \cdot b_i) \bmod [(m_i - 1)/2]\}' = \{[(\gamma_{a_i}, a_i) \times \\ \times (\gamma_{b_i}, b_i)] \bmod [(m_i - 1)/2]\}' = (a_i' \cdot b_i') \bmod m_i + \\ + (m_i - 1)/2 = \{[\gamma'_{a_i}, (a_i')^*] \cdot [\gamma'_{b_i}, (b_i')^*]\} \bmod m_i + \\ + (m_i - 1)/2 = \{\gamma_i', [(a_i')^* \cdot (b_i')^*] \bmod m_i + (m_i - 1)/2; \\ \{(a_i \cdot b_i) \bmod [(m_i - 1)/2]\}' = \{[(\gamma_{a_i}, a_i) \times \\ \times (\gamma_{b_i}, b_i)] \bmod [(m_i - 1)/2]\}' = (a_i' \cdot b_i') \bmod m_i = \\ = \{[\gamma'_{a_i}, (a_i')^*] \cdot [\gamma'_{b_i}, (b_i')^*]\} \bmod m_i = \\ = \{\gamma_i', [(a_i')^* \cdot (b_i')^*] \bmod m_i\}. \end{cases} \tag{24}$$

Для m_i – парного числа отримаємо

$$\begin{cases} (a_i \cdot b_i) \bmod [m_i / 2]\}' = \{[\,(\gamma_{a_i}, a_i) \times \\ \times (\gamma_{b_i}, b_i)\,] \bmod [m_i / 2]\}' = \\ = (a_i' \cdot b_i') \bmod m_i + m_i / 2 = \{[\,\gamma'_{a_i}, (a_i')^*\,] \times \\ \times [\gamma'_{b_i}, (b_i')^*\,]\} \bmod m_i + m_i / 2 = \{\gamma'_i, (a_i')^* \times \\ \times (b_i')^*\,]\} \bmod m_i + m_i / 2; \\ \{(a_i \cdot b_i) \bmod [m_i / 2]\}' = \{[\,(\gamma_{a_i}, a_i) \cdot (\gamma_{b_i}, b_i)] \bmod [m_i / 2]\}' = \\ = (a_i' \cdot b_i') \bmod m_i = \{[\gamma'_{a_i}, (a_i')^*] \cdot [\gamma'_{b_i}, (b_i')^*]\} \bmod m_i = \\ = \{\gamma'_i, [\,(a_i')^* \cdot (b_i')^*\,] \bmod m_i. \end{cases} \quad (25)$$

Співвідношення (24) ÷ (25) є математичною моделлю процесу табличної реалізації операцій алгебраїчного множення в КЗ.

4 Висновки

Таким чином, на підставі властивостей КЗ в статті синтезована математична модель процесу табличної реалізації модульного множення, як для позитивного, так і для негативного числових діапазонів обробки інформації ЗОІ. Дана модель рекомендована до практичного застосування при розробці методів і алгоритмів швидкої обробки криптографічної інформації.

Посилання

[1] Akushsky I. Ya.,Yuditsky DI. Machine arithmetic in residual classes, Moscow: Soviet radio, 1968, 440 p. (In Russian)

[2] Torgashov V. A. The system of residual classes and reliability of digital computers, Moscow: Soviet radio, 1973, 118 p. (In Russian)

[3] Krasnobayev V., Kuznetsov A., Koshman S., Moroz S. (2019) Improved Method of Determining the Alternative Set of Numbers in Residue Number System. In: Chertov O., Mylovanov T., Kondratenko Y., Kacprzyk J., Kreinovich V., Stefanuk V. (eds) Recent Developments in Data Science and Intelligent Analysis of Information. ICDSIAI 2018. Advances in Intelligent Systems and Computing, vol 836. Springer, Cham, pp. 319-328, 05 August 2018.

[4] F. Barsi and P. Maestrini, "Error Correcting Properties of Redundant Residue Number Systems," in IEEE Transactions on Computers, vol. C-22, no. 3, pp. 307-315, March 1973.

[5] V.A. Krasnobayev, A.S. Yanko, S.A. Koshman. “A Method for arithmetic comparison of data represented in a residue number system” Cybernetics and Systems Analysis, vol. 52, issue 1, pp. 145-150, January 2016.

[6] G. Harman and I. E. Shparlinski, "Products of Small Integers in Residue Classes and Additive Properties of Fermat Quotients," in International Mathematics Research Notices, vol. 2016, no. 5, pp. 1424-1446, Jan. 2016.

[7] V. Krasnobayev, A. Kuznetsov, M. Zub, K. Kuznetsova. Methods for comparing numbers in non-positional notation of residual classes. In Proceedings of the Second International Workshop on Computer Modeling and Intelligent Systems (CMIS-2019), Zaporizhzhia, Ukraine, April 15-19, 2019., pp. 581–595. 2019.

[8] D. I. Popov and A. V. Gapochkin, "Development of Algorithm for Control and Correction of Errors of Digital Signals, Represented in System of Residual Classes," 2018 International Russian Automation Conference (RusAutoCon), Sochi, 2018, pp. 1-3.

[9] Yu.V. Stasev, A.A. Kuznetsov, A.M. Nosik. “Formation of pseudorandom sequences with improved autocorrelation properties.” Cybernetics and Systems Analysis, vol. 43, Issue 1, pp. 1-11, January 2007.

[10] A. Kuznetsov, A. Kiian, K. Kuznetsova, et al. Soft decoding based on ordered subsets of verification equations of turbo-productive codes. In Proceedings of the Second International Workshop on Computer Modeling and Intelligent Systems (CMIS-2019), Zaporizhzhia, Ukraine, April 15-19, 2019., pp. 873–884. 2019

[11] A. A. Kuznetsov, I. P. Kolovanova, D. I. Prokopovych-Tkachenko, T. Y. Kuznetsova. “Analysis and Studying of the Properties of Algebraic Geometric Codes.” Telecommunications and Radio Engineering, Volume 78, 2019, Issue 5, pp. 393-417.

[12] B. Wang and L. Liu, "A flexible and energy-efficient reconfigurable architecture for symmetric cipher processing," 2015 IEEE International Symposium on Circuits and Systems (ISCAS), Lisbon, 2015, pp. 1182-1185.

[13] A. Kuznetsov, Y. Gorbenko, A. Andrushkevych and I. Belozersev, "Analysis of block symmetric algorithms from international standard of lightweight cryptography ISO/IEC 29192-2," 2017 4th International Scientific-Practical Conference Problems of Infocommunications. Science and Technology (PIC S&T), Kharkov, 2017, pp. 203-206.

[14] I. Gorbenko, A. Kuznetsov, M. Lutsenko and D. Ivanenko, "The research of modern stream ciphers," 2017 4th International Scientific-Practical Conference Problems of Infocommunications. Science and Technology (PIC S&T), Kharkov, 2017, pp. 207-210.

[15] M. E. Pamukov, V. Poulkov, A. Mihovska, N. R. Prasad and R. Prasad, "Lightweight robust cryptographic combiner for mobile devices: Crypto roulette," 2014 IEEE 19th International Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD), Athens, 2014, pp. 188-192.

[16] A. Kuznetsov, I. Kolovanova and T. Kuznetsova, "Periodic characteristics of output feedback encryption mode," 2017 4th International Scientific-Practical Conference Problems of Infocommunications. Science and Technology (PIC S&T), Kharkov, 2017, pp. 193-198.

[17] Z. Dai, X. Yu, J. Su and X. Chen, "Accelerated Flexible Processor Architecture for Crypto Information," 2007 2nd International Conference on Pervasive Computing and Applications, Birmingham, 2007, pp. 399-403.

[18] I. Gorbenko, O. Kuznetsov, Y. Gorbenko, A. Alekseychuk and V. Tymchenko, "Strumok keystream generator," 2018 IEEE 9th International Conference on Dependable Systems, Services and Technologies (DESSERT), Kyiv, Ukraine, 2018, pp. 294-299.

[19] Runovski, K., & Schmeisser, H. (2004). On the convergence of fourier means and interpolation means. Journal of Computational Analysis and Applications, 6(3), 211-227.

[20] Gnatyuk, V. A. (2001). Mechanism of laser damage of transparent semiconductors. Physica B: Condensed Matter, 308-310, 935-938.

[21] Tkach, B. P., & Urmancheva, L. B. (2009). Numerical-analytic method for finding solutions of systems with distributed parameters and integral condition. Nonlinear Oscillations, 12(1), 113-122.

[22] Chornei, R., Hans Daduna, V. M., & Knopov, P. (2005). Controlled markov fields with finite state space on graphs. Stochastic Models, 21(4), 847-874.

[23] Y. N. Kocherov, D. V. Samoylenko and A. I. Koldaev, "Development of an Antinoise Method of Data Sharing Based on the Application of a Two-Step-Up System of Residual Classes," 2018 International Multi-Conference on Industrial Engineering and Modern Technologies (FarEastCon), Vladivostok, 2018, pp. 1-5.

[24] C. Fan and G. Ge, "A Unified Approach to Whiteman's and Ding-Helleseth's Generalized Cyclotomy Over Residue Class Rings," in IEEE Transactions on Information Theory, vol. 60, no. 2, pp. 1326-1336, Feb. 2014.

[25] M. Kasianchuk, I. Yakymenko, I. Pazdriy, A. Melnyk and S. Ivasiev, "Rabin's modified method of encryption using various forms of system of residual classes," 2017 14th International Conference The Experience of Designing and Application of CAD Systems in Microelectronics (CADSM), Lviv, 2017, pp. 222-224.

[26] V.A. Krasnobayev, S.A. Koshman, M.A. Mavrina. "A Method for Increasing the Reliability of Verification of Data Represented in a Residue Number System" Cybernetics and Systems Analysis, , vol. 50, issue 6, pp. 969–976, November 2014.

[27] K. Tao, L. Peng, K. Liang and B. Zhuo, "Irregular repeat accumulate low-density parity-check codes based on residue class pair," 2017 IEEE 9th International Conference on Communication Software and Networks (ICCSN), Guangzhou, 2017, pp. 127-131.

[28] V. Krasnobayev, A. Kuznetsov, A. Kononchenko, T. Kuznetsova. Method of data control in the residue classes. In Proceedings of the Second International Workshop on Computer Modeling and Intelligent Systems (CMIS-2019), Zaporizhzhia, Ukraine, April 15-19, 2019., pp. 241–252. 2019.

[29] V. Krasnobayev, S. Koshman, A. Yanko and A. Martynenko, "Method of Error Control of the Information Presented in the Modular Number System," 2018 International Scientific-Practical Conference Problems of Infocommunications. Science and Technology (PIC S&T), Kharkiv, Ukraine, 2018, pp. 39-42.

[30] Krasnobayev V. A. Method for realization of transformations in public-key cryptography. Telecommunications and Radio Engineering. - Volume 66, 2007 Issue 17, pp. 1559-1572.

Reviewer: Alexandr Kuznetsov, Doctor of Sciences (Engineering), Full Prof., Academician of the Academy of Applied Radioelectronics Sciences, V.N. Karazin Kharkiv National University, Kharkov, Ukraine.
E-mail: kuznetsov@karazin.ua

Received: September 2019.

Authors:

Viktor Krasnobayev, Doctor of Sciences (Engineering), Full Prof., Academician of the Academy of Applied Radioelectronics Sciences, V.N. Karazin Kharkiv National University, Kharkov, Ukraine.
E-mail: v.a.krasnobaev@gmail.com

Mikhail Zub, student, V.N. Karazin Kharkiv National University, Kharkov, Ukraine.
E-mail: mishazub007@gmail.com

Olesya Reshetnyak, student, V.N. Karazin Kharkiv National University, Kharkov, Ukraine.
E-mail: lesyandr13@gmail.com

Andrey Dyachenko, student, V.N. Karazin Kharkiv National University, Kharkov, Ukraine.
E-mail: andrey.090220@gmail.com

Irina Lokotkova, student, V.N. Karazin Kharkiv National University, Kharkov, Ukraine.
E-mail: lokotosyk@ukr.net

Konstantin Myslyvtsev, student, V.N. Karazin Kharkiv National University, Kharkov, Ukraine.
E-mail: kostyavtx@gmail.com

Mathematical model of the process of tabular's implementation of the operation algebraic multiplication in the residues class.

Abstract. Based on the properties of the class of residues in the article, a mathematical model of the process of tabular implementation of modular multiplication was synthesized, for both positive and negative numerical ranges of information processing by means of integer information processing. This model is recommended for practical application in the development of methods and algorithms for rapid processing of cryptographic information. Finding ways to simplify the structure of the table operating device integer information processing has led to the need to improve the mathematical model, methods and algorithms for modular operations, which increase the efficiency of the use of table arithmetic in the class of residues. A feature of the implementation of this model is the possibility of reducing the number of equipment operating device integer information by reducing the (50-70)% of logical ele-

ments "And" in the nodes of the table permanent storage device, directly implement the operation of modular multiplication by arbitrary m_i module residual class. This is possible by using the symmetry properties of the a_ib_i (mod m_i) implementation table of the modular multiplication operation.

Keywords: Residues Class; Tabular's Implementation; Mathematical Model; Operation Algebraic Multiplication.

Рецензент: Александр Кузнецов, д.т.н., проф., академик Академии наук прикладной радиоэлектроники, Харьковский национальный университет им. В.Н. Каразина, Харьков, Украина.
E-mail: kuznetsov@karazin.ua

Поступила: Сентябрь 2019.

Авторы:

Виктор Краснобаев, д.т.н., проф., академик Академии наук прикладной радиоэлектроники, Харьковский национальный университет им. Каразина.
E-mail: v.a.krasnobaev@gmail.com

Михаил Зуб, студент факультета компьютерных наук, Харьковский национальный университет им. Каразина.
E-mail: mishazub007@gmail.com

Олеся Решетняк, студент факультета компьютерных наук, Харьковский национальный университет им. Каразина.
E-mail: lesyandr13@gmail.com

Андрей Дьяченко, студент факультета компьютерных наук, Харьковский национальный университет им. Каразина.
E-mail: andrey.090220@gmail.com

Ирина Локоткова, студентка факультета компьютерных наук, Харьковский национальный университет им. Каразина.
E-mail: lokotosyk@ukr.net

Константин Мисливцев, студент факультета компьютерных наук, Харьковский национальный университет им. Каразина.
E-mail: kostyavtx@gmail.com

Математическая модель процесса табличной реализации операции алгебраического умножения в классе вычетов.

Аннотация. На основании свойств класса вычетов в статье синтезирована математическая модель процесса табличной реализации модульного умножения, как для положительного, так и для отрицательного числовых диапазонов обработки информации средствами обработки целочисленной информации. Данная модель рекомендована к практическому применению при разработке методов и алгоритмов быстрой обработки криптографической информации. Поиск путей упрощения структуры табличного операционного устройства средства обработки целочисленной информации обусловил необходимость совершенствования математической модели, методов и алгоритмов реализации модульных операций, позволяющих повысить эффективность применения табличной арифметики в классе остатков. Особенностью реализации данной модели является возможность уменьшения количества оборудования операционного устройства средства обработки целочисленной информации за счет сокращения на (50-70)% логических элементов "И" в узлах таблицы постоянное запоминающее устройство, непосредственно реализующие операцию модульного умножения по произвольному m_i модулю класса остатков. Это возможно за счет использования свойств симметрии таблицы реализации a_ib_i (mod m_i) модульной операции умножения.

Ключевые слова: класс вычетов; табличная реализация; математическая модель; операция алгебрическое умножение.

УДК 004.056.55 DOI: 10.26565/2519-2310-2019-2-02

PARALLELIZATION OF THE MATHEMATICAL MODEL OF GABOR FILTER

Serghii Rassomakhin, Artur Bykov

V. N. Karazin Kharkiv National University, 6 Svobody Sq., Kharkiv, 61022, Ukraine
rassomakhin@karazin.ua, bykov.artur.kh.96@gmail.com

Reviewer: Vyacheslav Kalashnikov, Doctor of Sciences (Physics and Mathematics), Full Prof., Department of Systems and Industrial Engineering, Campus Monterrey, Monterrey, 64849, Mexico.
kalash@itesm.mx

Received: September 2019

***Abstract**: Due to the acceleration of the informatization of modern society and the increase in the number of flows and objects of information that must be protected from unauthorized access, problems with the use of biometric identification technologies to differentiate access to information resources are becoming increasingly important. The paper discusses some features of fingerprint preprocessing procedures using the Gabor filter. A block diagram of a parallel model of software for fingerprint preprocessing is presented. Various technologies are also briefly considered that make it possible to organize an appropriate program (parallel processing), the advantages and disadvantages of each of them are described, on the basis of which the optimal one is selected for a specific task. The results on the speed of program execution with a different number of threads and their priority are provided.*

***Key words: fingerprints**: Biometric Image; Gabor Function; Parallel Execution*

1 Introduction

Due to the increasing informatization of modern society and the increasing number of flows and objects of information that need to be protected from unauthorized access, problems with the use of biometric identity identification technologies to differentiate access to information resources are becoming more urgent. The use of biometric characteristics to confirm an identity involves the use of physical characteristics, such as voice or fingerprints, for identification purposes. Fingerprint mapping is the most successful biometric identification technology due to its ease of use, non-interference and reliability. The fingerprint consists of furrows and stripes, forming a complex pattern unique to each person, and therefore provide the optimum method of verification. Usually, the problem with fingerprint identification, namely fingerprints, is poor image or fuzzy lines. The subject matter of the development is an information and technical system for the processing of fingerprint images. The problem they solve is that it is necessary to improve the image of fingerprints for further analysis of fingerprints.

2 The Gabor filter and implementation of the software product

The application of the Gabor filter enhances the image. It is contrasted with the lines of prints, noise is removed. Binarized images are convenient for further conversion: skeletonization and search for special points on prints [1-6]. Using coherence allows you to highlight the area of interest in which additional calculations are performed. The disadvantage of the algorithm is the representation of a curve line as a set of directions in each segment, which can cause a fuzzy response of the filter in the areas and curvatures and curls. On the other hand, the incremental sampling associated with the decrease in segment size entails a large margin of error in determining the average line orientation of each segment.

Gabor filter transfer function

$$G(x,y,\varphi)=e^{\left(\frac{-\frac{1}{2}\left(x\cos(\varphi)+y\sin(\varphi)\right)^2}{\sigma x^2}+\frac{\left(-x\sin(\varphi)+y\cos(\varphi)\right)^2}{\sigma y^2}\right)}\cos\left(2\pi\theta\left(x\cos(\varphi)+y\sin(\varphi)\right)\right)-Z.. \quad (1)$$

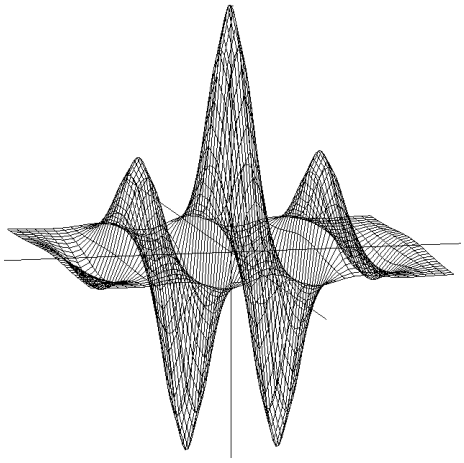


Fig. 1 – Gabor function

The C^{++} programming language was chosen to implement the software. C^{++} is a compiled, statically typed general-purpose programming language. Supports programming paradigms such as procedural programming, object-oriented programming, generalized programming. The language has a rich standard library, which includes common containers and algorithms, I/O, regular expressions, lots of accuracy support, and other features. C^{++} combines the properties of both high-level and low-level languages. The advantage of written software is the lack of third-party libraries. The program is based solely on standard tools provided by Microsoft. As well as minimizing the use of objects for easy translation of the program into the C programming language, as needed. The program has no memory leaks and handlers (*repeated testing*).

In order to write a program that in turn will use the computing power of the computer as efficiently as possible. The main technologies for parallelizing software, namely the Message Passing Interface (MPI), OpenMP (Open Multi-Processing) and POSIX Threads, were reviewed. Each technology has a number of advantages and disadvantages.

MPI (message passing interface) - the transmission system is standardized (function library). The standard is the syntax and semantics of the current functions, use when writing portable programs with the transmission of translations on the moves Fortran 77, C and C^{++}. The main way of sharing parallel processes in a given whistle is to transfer a single site to a single site. The MPI standard is the interface, which is guilty of being impaired as a system by the program on the leather calculating platform, so we are corrupted when setting up our programs [7]. MPI-program - with no parallel processes. All processes are done once, asserting parallel part of the programs. Each process works in its address space, there are no common variables or data in MPI. The main way to interact with the processes є is to make sure that one of the processes is completed.

It's independent of those that use MPI programs to show the highest level of productivity, the technology itself has a number of short-lived [7]:

- Low level (programming on the MPI is often the same as programming on the assembler), the need for detailed control of the output and the number of cycles, as well as the greater the number of processes — all the more important;
- Necessity of superlative specificity of types in tributes, as well as the very manifestation of hardcore divisions in tipi of transferred tributes;
- Flexibility of writing programs, keeping your eyes peeled at the end of the week of high-tech and high-quality processes - to rob is practically impractical to re-enter the wake-up MPI-programs;
- Visibility of the object-oriented approach.

OpenMP promotes a simple mechanism for realizing parallel parallels in additions for additional multithreading, in the "master" thread and the "slave" thread, and the redundancy of them. OpenMP standard for Fortran, C / C^{++} (*Fortran is not to be looked at*) and for formulating an API for writing sterile bagging threading on multiprocessor systems using the Single Program Multiple Data (SPMD) [8]. Portability is tied to the OpenMP program model, which requires an independent platform for compiling directives (pragmas), functions and middleware, obviously showing a compiler, and I'm talking about parallelism. For a programmer, there's no need for additional storage, for getting things done, for synchronizing, for balancing and for getting things done. OpenMP allows you to see how it's possible over Pthreads, the POSIX interface for organizing threads. At Pthreads itself - it's a busy low level of programmability, a little bit of parallelism for giving, and the very

mechanism of flowing to bed is not for the purposes of organizing parallelism. OpenMP has terminology and programming model, close to Pthreads, for example, dynamically generated threads, external and collective data, etc. OpenMP technology is aimed at those who want to use one version of the program for a parallel and last weekend. However, it's possible to set programs so that you can correct ones in parallel mode, but give the last result in the last mode. In addition, through the accumulation of forgiveness rounded off, the result of the weekend shall be deducted from the cities of the Republic of Kazakhstan in possible deceases.

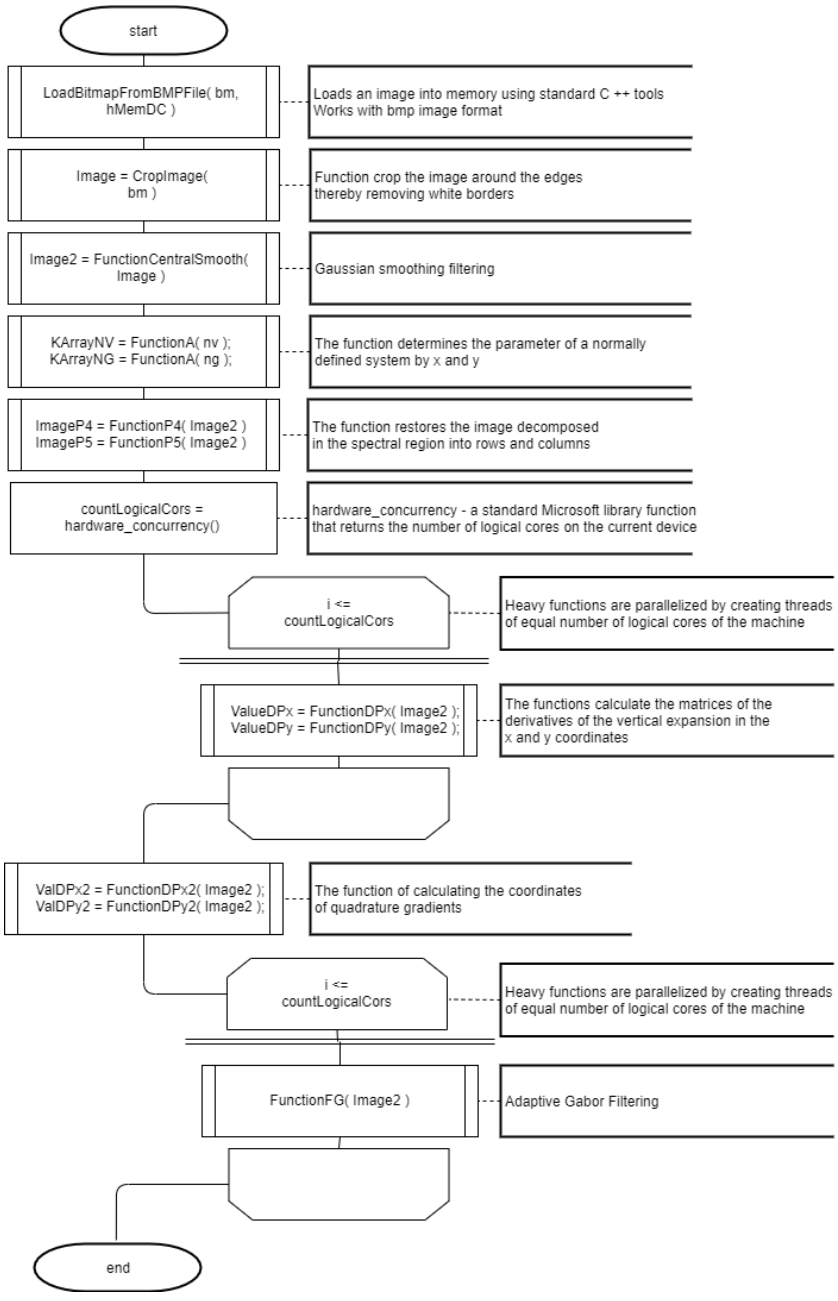


Fig. 2 – Block diagram

POSIX Threads, usually referred to as pthreads, is an execution model that exists independently from a language, as well as a parallel execution model. It allows a program to control multiple different flows of work that overlap in time. Each flow of work is referred to as a thread, and creation and control over these flows is achieved by making calls to the POSIX Threads API. POSIX Threads is an API defined by the standard POSIX.1c, Threads extensions (IEEE Std 1003.1c-1995) [9].

Before performing the Gabor function on the image, a number of manipulations are necessary [1,3]. Some of them are quite complex and also need to be parallelized. Since the process of creating and starting the stream is quite heavy, the optimal solution for parallelizing the program was found. Figure 2 shows a block diagram of the main function which describes the main stages of the work progress.

3. Program execution speed testing

Product testing was conducted on a computer HP 4540s Intel i5 processor - 3230M CPU 2.6GHz. This processor model contains 4 logical cores, which made it possible to fully test the speed of a program with a different number of threads.

Table. 1 – Program speed results

Thread Count / Priority	1	2	3	4
NORMAL	47s	39s	31s	28s
HIGHEST	45s	30s	23s	20s

Figure 3 shows an example of an input image and Figure 4 shows an example of an image modified by a Gabor filter.

Fig.3 – Input image

Fig.4 – Modified image

3. Conclusions

A brief overview of the main procedures has been completed of fingerprint image preprocessing by the Gabor filter. Having also analyzed the most common technologies for organizing parallel execution of processes available in the C^{++} programming language. Taking into account the fact that the mathematical model can be divided into subtasks that will not be interconnected, that is, the principles of data locality are preserved - we can conclude that there are no advantages for this task in the mpi technology. The next technology on the list is Open MP, in which the model is pretty close to our task (*processing of fingerprint image*), but it also has a number of disadvantages, the main one being the fact that the Microsoft C^{++} compiler only supports ver. 2.0, which does not have support for setting priorities for child threads, which is significant limits our possibilities. Therefore, for a specific task, POSIX Threads is the preferred option that encapsulates fairly quick and convenient thread manipulations.

Посилання

[1] Гудков В. Ю., Бойцов А. В. Улучшение изображений отпечатков пальцев с помощью фильтра Габора. URL: https://cyberleninka.ru/article/n/uluchshenie-izobrazheniy-otpechatkov-paltsev-s-pomoschyu-filtra-gabora (дата звернення 05.09.2019).

[2] Костенко Л. С. Методы и алгоритмы сглаживания фона изображений в системах распознавания образов / Л. С. Костенко. Открытые информационные и компьютерные интегрированные технологии. 2014. Вып. 64. С. 177-181. URL: http://nbuv.gov.ua/UJRN/vikt_2014_64_21 (дата звернення 03.09.2019).

[3] Ляхов П. А. Валуева М. В. Применение сглаживающих фильтров для очистки от шума изображений в оттенках серого. Наука. Инновации. Технологии. 2015. №3. URL: https://cyberleninka.ru/article/n/primenenie-sglazhivayuschih-filtrov-dlya-ochistki-ot-shuma-izobrazheniy-v-ottenkah-serogo (дата звернення 05.09.2019).

[4] Федюкович Г. Г. Сравнение использования технологий параллельного программирования Microsoft применительно к задаче поиска данных MapReduce URL: http://omega.sp.susu.ru/books/conference/PaVT2010/poster/169.pdf (дата звернення 05.09.2019).

[5] Гудков В. Ю., Бойцов А. В. Улучшение изображений отпечатков пальцев с помощью фильтра Габора. URL: https://cyberleninka.ru/article/n/uluchshenie-izobrazheniy-otpechatkov-paltsev-s-pomoschyu-filtra-gabora (дата звернення 05.09.2019).

[6] Рыканов А.С. Анализ методов распознания отпечатков пальца. Системи обробки інформації. 2010. Вип.6 (87) . С.164 – 171. URL: http://www.hups.mil.gov.ua/periodic-app/article/7838/soi_2010_6_37.pdf (дата звернення 05.09.2019).

[7] MPI: A Message-Passing Interface Standard Version 3.0. URL: https://www.mpi-forum.org/docs/mpi-3.0/mpi30-report.pdf (дата звернення 25.09.2019).

[8] Параллельное программирование на OpenMP. URL: http://ccfit.nsu.ru/arom/data/openmp.pdf (дата звернення 11.10.2019).

[9] Blaise B. POSIX Thread – Introduction to multithreading. URL: https://computing.llnl.gov/tutorials/pthreads/ (дата звернення 07.09.2019).

Рецензент: В'ячеслав Калашников, д.ф.-м.н., проф., Технологический університет Монтеррея, пр. Еухеніо Гарса Сада 2501, Монтеррей, 64849, Мексика.
E-mail: kalash@itesm.mx

Надійшло: Вересень 2019.

Автори:
Артур Биков, студент факультету комп'ютерних наук, ХНУ ім. В.Н. Каразіна, майдан Свободи 4, м. Харків, 61022, Україна.
E-mail: bykov.artur.kh.96@gmail.com
Сергій Рассомахін, д.т.н., проф., ХНУ ім. В. Н. Каразіна, майдан Свободи 6, Харків, 61022, Україна.
E-mail: rassomakhin@karazin.ua

Розпаралелювання математичної моделі фільтра Габора.

Анотація. У зв'язку з прискоренням інформатизації сучасного суспільства та збільшенням числа напрямків і об'єктів інформації, які необхідно захистити від несанкціонованого доступу, проблеми з використанням технологій біометричної ідентифікації особистості для розмежування доступу до інформаційних ресурсів стають дедалі актуальнішими. В роботі розглядаються деякі особливості процедур попередньої обробки відбитків пальців з використанням фільтра Габора. Представлена структурна схема паралельної моделі програмного забезпечення для попередньої обробки відбитків пальців. Також коротко розглянуті різні технології, що дозволяють організувати відповідну програму (паралельної обробки), описуються переваги і недоліки кожної з них, на основі яких вибирається оптимальна для конкретного завдання. Надано результати по швидкості виконання програм з різною кількістю потоків і їх пріоритетом.

Ключові слова: відбитки пальців; біометричне зображення; функція Габора; паралельне виконання.

Рецензент: Вячеслав Калашников, д.т.н., проф., Технологический университет Монтеррея, пр. Еухенио Гарса Сада 2501, Монтеррей, 64849, Мексика.
E-mail: kalash@itesm.mx

Поступила: Сентябрь 2019.

Авторы:
Артур Быков, студент факультета компьютерных наук, ХНУ им. В. Н. Каразина, пл. Свободы, 4, Харьков, 61022, Украина.
E-mail: bykov.artur.kh.96@gmail.com
Сергей Рассомахин, д.т.н., проф., ХНУ им. В. Н. Каразина, пл. Свободи 6, Харьков, 61022, Украина.
E-mail: rassomakhin@karazin.ua

Распараллеливание математической модели фильтра Габора.

Аннотация. В связи с ускорением информатизации современного общества и увеличением числа направлений и объектов информации, которые необходимо защитить от несанкционированного доступа, проблемы с использованием технологий биометрической идентификации личности для разграничения доступа к информационным ресурсам становятся все более актуальными. В работе рассматриваются некоторые особенности процедур предварительной обработки отпечатков пальцев с использованием фильтра Габора. Представлена структурная схема параллельной модели программного обеспечения для предварительной обработки отпечатков пальцев. Также коротко рассмотрены различные технологии, позволяющие организовать соответствующую программу (параллельной обработки), описываются преимущества и недостатки каждой из них, на основе которых выбирается оптимальная для конкретной задачи. Предоставлены результаты по скорости выполнения программ с разным количеством потоков и их приоритетом.

Ключевые слова: отпечатки пальцев; биометрическое изображение; функция Габора; параллельное исполнение.

УДК 004.438 DOI: 10.26565/2519-2310-2019-2-03

ДОСЛІДЖЕННЯ КРОСПЛАТФОРМНОГО ФРЕЙМВОРКУ FLUTTER. ЧИ ОЗНАЧАТИМЕ РОЗКВІТ ЦІЄЇ ТЕХНОЛОГІЇ ЗНИКНЕННЯ НАТИВНОЇ РОЗРОБКИ НА ANDROID ТА iOS?

Олександр Синельніков

Харківський національний університет імені В.Н. Каразіна, майдан Свободи 4, Харків, 61022, Україна
pride.sin.al@gmail.com,

Рецензент: Олександр Потій, д.т.н., проф., Харківський національний університет імені В.Н. Каразіна, майдан Свободи 6, Харків, 61022, Україна
potav@ua.fm

Поступила: жовтень 2019

Анотація. *Робота присвячена дослідженню кросплатформного фреймворку для розробки мобільних додатків Flutter від компанії Google. Проведено практичне випробування технології з ціллю встановити, чи вона достатньо зручна та надійна. У статті представлені висновки за результатами проведених досліджень. Наведено порівняння механізмів роботи Flutter та інших популярних мобільних кросплатформних фреймворків. Розглянуто основні особливості і відмінності технології. Надано рекомендації щодо її використання.*

Ключевые слова: *кросплатформні рішення; мобільні додатки; нативна розробка; мова програмування Dart.*

1 Введение

Станом на сьогодні статистика використання мобільних додатків (МД) говорить про те, що ринок зростає з геометричною прогресією, а особливо стрімка тенденція простежується саме серед кросплатформних програмних рішень, які набувають все більшої популярності. Мова йде про програмне забезпечення, яке може запускатися на різних платформах (*Android та iOS*), маючи при цьому одну кодову базу. Перевага таких МД полягає у зниженні витрат та прискоренні процесу їх розробки. Це робить крос платформні (К-П) програмні рішення дуже привабливими, особливо, у корпоративному секторі інформаційних технологій. Та чому ж попри всі переваги, нативна розробка (тобто розробка під одну конкретну платформу) МД і досі все ще затребувана? Справа в тому, що програмні рішення, на основі яких реалізовано К-П розробку, мають певній перелік недоліків, через які такі додатки поступаються нативним за тими чи іншими показниками (*наприклад, швидкість візуалізації елементів на екрані мобільного пристрою, продуктивність та інші*). У даної роботі представлений огляд К-П фреймворку Flutter, проведено його порівняння з іншими подібними технологіями, а також надано резюме досліду стосовно особливостей його практичного використання.

2 Принцип роботи Flutter та порівняння його з іншими рішеннями.

Flutter – це SDK (Software Development Kit) з відкритою кодовою базою для створення мобільних додатків від компанії Google. Він використовується для розробки додатків під Android та iOS, а також це основний спосіб розробки додатків під Google Fuchsia – оперативної системи від Google, яка зараз (станом на листопад 2019 року) знаходиться в стадії активної розробки; прогнозується, що вона дебютує через два роки і в кінцевому підсумку замінить Android, Wear OS і Chrome OS. Творці фреймворку стверджують, що він позбавлений більшості недоліків своїх попередників, а саме таких К-П фреймворків, як React Native, PhoneGap, Ionic, Xamarin та інших.

Ні для кого не секрет, що у сфері розробки К-П мобільних додатків є достатнє багатство вибору, що у різних технологій є свої переваги і недоліки. Що ж такого революційного може запропонувати нам Flutter? Щоб це наочно виявити, необхідно порівняти деталі архітектури Flutter та React Native. Справа в тому, що React використовує нативні віджети для відтворення UI і комунікацією з платформними компонентами. Він використовує для цього, так зва-

ний міст (див. Рис. 1), який і є тим «вузьким місцем», що значно уповільнює малювання (*візуалізації*) UI.

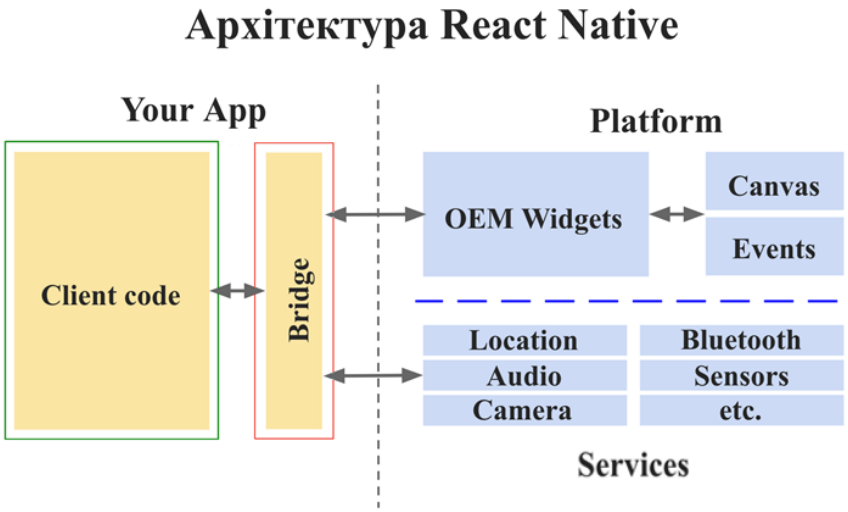


Рис. 1 – Схема архітектури React Native

В архітектурі Flutter (*флаттер*) такий міст відсутній. І хоча інтерфейс між Dart і платформним кодом все ще існує (Рис. 2), йому для роботи потрібно значно менше часу.

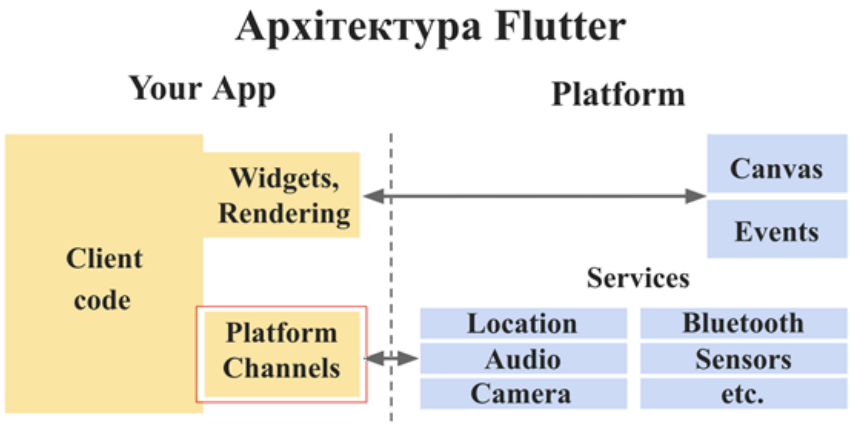


Рис. 2 – Схема архітектури Flutter

Варто звернути увагу на малювання елементів. У кросплатформних рішеннях, які використовують нативні компоненти UI, для взаємодії з ними використовується віртуальне дерево віджетів (Рис. 3).

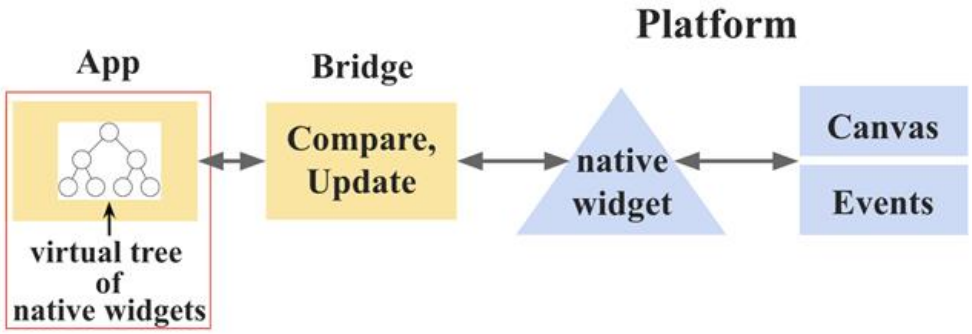


Рис. 3 – Схема малювання елементів у React Native

Віртуальне дерево – це абстрактна версія зображення з екрану, створена з використанням об'єктів, тобто програмне уявлення того, що бачить користувач. Віртуальне дерево є незмінним і перебудовується з нуля кожен раз, коли що-небудь треба поміняти на екрані. Віртуальне дерево порівнюється з положенням і станом нативних віджетів для генерації набору мінімальних змін, які потім виконуються, щоб оновити стан. Нарешті, платформа повторно відображає реальне дерево і малює його на екрані. Все це необхідно лише тому, що оновлення всього екрану відразу – занадто дорога в сенсі продуктивності операція.

Однак Flutter не використовує нативні елементи екрану або навіть WebViews, він надає свої власні віджети. Це збільшує їх гнучкість та швидкість рендеринга. Flutter малює (відображує) свої віджети прямо на canvas. Таким чином, те, що було віртуальним деревом віджетів в попередньому прикладі, тепер є безпосередньо деревом віджетів (Рис. 4), яке відображується на екрані.

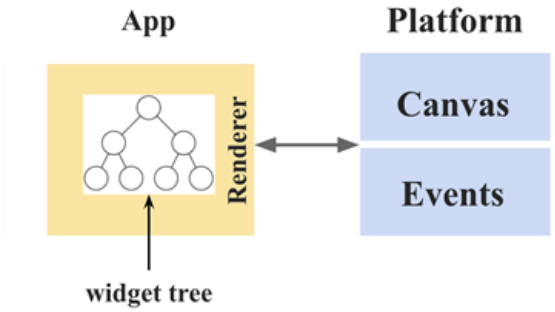


Рис. 4 – Схема відображення елементів у Flutter

Рендерер Flutter перемальовує тільки ті віджети, які необхідно оновити на екрані. Незмінені віджети, навіть ті, які були просто переміщені, частково відновлюються з кешу (дуже швидко).

3 Мова програмування Dart

Dart – це мова програмування з повністю відкритим вихідним кодом, саме на ній працює Flutter. «Всередині» Google ця мова розвивається дуже швидко, вона використовується Adwords, Flutter, Fuchsia та іншими. Нижче представлений невеликий перелік деяких принципових особливостей Dart, які значно спрощують роботу з Flutter:

- Dart може бути AOT (Ahead Of Time) компільованим в швидкий, передбачуваний, машинний ARM код. Це робить Flutter дуже швидким в плані роботи (для збірки release).
- Dart може бути JIT (Just In Time) компільованим для виключно швидких циклів розробки (*включаючи популярну Hot reload Flutter, про яку мова піде трохи пізніше*).
- Код в Дарт є однопоточним, що позбавляє розробників від певних проблем синхронізації даних. При цьому, якщо все ж виникає необхідність у багатопоточності, то є відповідні ізоляти (*isolates*), які виконують роль потоків, і працюють таким чином, що у них немає області спільної пам'яті, а спілкування між ними відбувається через відправку повідомлень по спеціальним каналам.
- Схема «збору сміття» і розподілу пам'яті в Dart особливо швидка для розподілу пам'яті для безлічі недовговічних об'єктів (*ідеально підходить для Flutter, який знищує та повторно перебудовує незмінне дерево віджетів для кожного кадру*).
- Хоча програмістів, які знають цю мову, поки ще, не так вже й багато, ті, хто знають мови Java, JavaScript, Kotlin, C# або Swift, вже можуть почати програмування на Dart практично відразу.

4 Особливості «Hot reload»

Варто звернути особливу увагу на одну з особливостей Flutter – Hot reload. Вона дозволяє розробникам вносити зміни в додаток в той час, коли він уже запущений, при цьому перезавантажиться тільки змінений код, а роботу з додатком можна продовжити з того ж місця, де

розробник зупинився до зміни коду. Весь описаний процес займає менше секунди, а все це можливо завдяки згаданому вище JIT-компілятору. Завдяки цьому функціоналу зникає необхідність в попередньому перегляді зовнішнього вигляду екрану, як це було, наприклад, з xml файлами в Java/Kotlin проектах. В цьому разі можна буквально створювати свій додаток прямо на екрані пристрою без необхідності перезапускати сам додаток.

5 Власний досвід розробки

Найпростіший спосіб знайти сильні та слабкі сторони будь-якого інструменту – спробувати щось зробити з його допомогою. Так, щоб з'ясувати чи є Flutter достатньо стабільний та зручний, та чи можна за його допомогою комфортно створювати додатки, було вирішено створити відповідний додаток для цілей бронювання кімнат для ділових зустрічей.

Творці флаттер подбали про міграцію розробників з інших платформ і пыдготовили велику кількість статей та відповідної документації. На офіційному сайті [1] можна знайти багато корисної інформації по розробці, а також матеріали щодо переходу з Android/iOS/Web розробки на флаттер. Так, первинне ознайомлення займає від пари днів до тижня. В результаті вже можна вільно орієнтуватися в простих прикладах коду фреймворку і розуміти Dart 2 код.

Одним з першочергових завдань був пошук сторонніх компонентів. Розробники Flutter подбали про зручність підключення бібліотек (у Flutter - плагінів). Список можливих залежностей можна знайти на спеціалізованому сайті [2], на якому викладені як офіційні плагіни, так і розроблені сторонніми розробниками (*в даному випадку можна провести аналогію з maven репозиторієм*). Плагіни можуть мати нативний код платформи, який дозволяє їм звертатися до компонентів системи. Такі плагіни під Android, створюються на Java або Kotlin, а під iOS – на Objective-C або Swift. Їх принцип роботи заснований на використанні каналів, де канали – це механізм зв'язку між Dart кодом та кодом конкретної платформи хост-додатку. Однак ці канали можуть передавати тільки бінарну і текстову інформацію. Таким чином, коли деякої нативної функціональності у флаттер немає, то розробник завжди має змогу надати її самостійно.

Враховуючи все вище зазначене, та спираючись на перші особисті враження від розробки тестового додатку, можна зробити наступні висновки стосовно плагінів:

1. Для стабільної роботи додатків необхідно ретельно підходити до вибору плагінів. В першу чергу варто дивитися на офіційні плагіни від розробників флаттер. Їх список є на github [3] і вони знаходяться прямо в репозиторії Flutter.
2. Якщо в запропонованому списку немає потрібного плагіна, то слід переміститися на сайт з плагінами від сторонніх розробників [2], на якому є безліч відповідних бібліотек. Але тут потрібно бути «обережним». Для стабільної роботи додатка варто вибирати тільки ті, для яких є не тільки приклади від розробників, а також і згадки на сторонніх ресурсах (*наприклад, питання з відповідями на StackOverflow*) [4].
3. Велика частина бібліотек, робоча. Однак, багато плагінів ще не мають стабільної версії. Будь-який з них може мати застарілу документацію або приклад використання. Також, хорошою порадою є перевірка працездатності плагіна на обох платформах.

6 Чи потрібен Flutter проекту?

Як зрозуміти що певний проект потрібно або, взагалі, можна стартувати на Flutter? В цілому, цей фреймворк варто використовувати коли завчасно відомий загальний обсяг роботи та відомо, що її можна реалізувати за допомогою стандартних засобів Flutter та плагінів. При цьому не будь-який додаток варто починати писати на Flutter. Так, прикладами поганих додатків на флатер слід вважати:

- додаток, який інтенсивно працює з нативними бібліотеками. На жаль, для того щоб викликати С-код, який нерідко необхідний в Android розробці, потрібно використовувати 2 моста. (Канали та JNI);

- якщо принципово важливе питання ефективного використання ресурсів. Наприклад, простий таймер буде використовувати більше пам'яті або потужності центрального процесору ніж нативний додаток;
- коли в додатку багато особливостей для яких доведеться створювати платформенний код. Наявність каналів робить можливим реалізувати на Flutter все, що можна зробити в нативному додатку, але писати одночасно 2 реалізації може виявитися занадто дорогим у сенсі накладних витрат. В цьому разі мова йде про такі особливості, як контент провайдер, CallKit і т.і.

У свою чергу, позитивними прикладами додатків на Flutter є:

- додаток, який має дизайн, що не залежить від платформи;
- додатки з складним інтерфейсом користувача. В цьому разі Flutter спрощує налаштування UI елементів;
- коли потрібна висока швидкість розробки. Адже необхідна лише одна команда та один клієнт-додаток.

5 Висновки

За результатами аналізу інформації профільних Інтернет ресурсів, узагальнення думки багатьох фахівців та особистого досвіду можна стверджувати, що Flutter є потенційно перспективною технологією, яка швидко набирає популярність. Кілька сміливих концепцій, що лежать в основі роботи даного фреймворку, привносять в розробку нові ідеї та можливості, завдяки яким створювати нові додатки стає значно простіше. Наприклад, таке поширене завдання, як вибір потрібного зображення зі сховища на мобільному пристрої у Flutter робиться одним рядком коду.

Flutter певно не зможе повністю замінити нативну розробку, але він точно задасть новий тренд, в якому в найближчі роки буде рухатися вагома частина індустрії мобільної розробки.

Стосовно перспектив розвитку фреймворку, команда Flutter оголосила про хід активної розробки Hummingbird – Flutter для Web – та розповіла, як це, приблизно, буде працювати.

В планах команди Flutter змінити лише самий низькорівневий шар таким чином, щоб код запускався в браузері. Як очікується, таким чином збережеться основна особливість Flutter – переносимість єдиної кодової бази на різні платформи, в даному випадку – на Web.

Більш детально результати даної роботи, було розглянуто в межах роботи конференції NixMulticonf, відео з якої можна знайти в переліку посилань [5].

Посилання

[1] Flutter dev. URL: https://flutter.dev/ (дата звернення 05.10.2019).
[2] Flutter packages. URL: https://pub.dev/flutter / (дата звернення 05.10.2019).
[3] Github flutter repository. URL: https://github.com/flutter/flutter / (дата звернення 05.10.2019).
[4] Stackoverflow. URL: https://ru.stackoverflow.com / (дата звернення 05.10.2019).
[5] Синельников А. Как приручить Flutter? URL: https://www.youtube.com/watch?v=fiwQy1vLvrM/ (дата звернення 05.10.2019).

Reviewer: Alexandr Potii, Dr. of Sciences (Engineering), Full Prof., V. N. Karazin Kharkiv National University, Kharkiv, 61022, Ukraine. E-mail: potav@ua.fm

Received: October 2019.

Authors:
Oleksandr Synelnikov, computer science student,, V. N. Karazin Kharkiv National University, Kharkiv, Ukraine.
E-mail: pride.sin.al@gmail.com

Investigation of the cross-platform Flutter framework. Will the heyday of this technology mean the disappearance of the native development on Android and iOS?

Abstract. The work is devoted to research of the cross-platform framework for developing mobile applications Flutter from Google. A practical test of the technology was carried out to establish whether it is convenient enough and reliable. The article presents the findings of the testing. The comparison of the mechanisms of Flutter and other popular mobile cross-platform frameworks is given. The main features and differences of the technology are considered. Recommendations on its use are formulated.

Keywords: Cross-Platform Solutions; Mobile Applications; Native Development; Programming Language Dart.

Рецензент: Александр Потий, д.т.н., проф., Харьковский национальный университет имени В. Н. Каразина, Харьков, 61022, Украина..
E-mail: potav@ua.fm

Поступила: Октябрь 2019.

Авторы:
Александр Синельников, студент факультета компьютерных наук, Харьковский национальный университет имени В.Н. Каразина, площадь Свободы, 4, Харьков, 61022, Украина.
E-mail: pride.sin.al@gmail.com

Исследование кросплатформенного фреймворка Flutter. Будет ли означать расцвет этой технологии исчезновение нативной разработки на Android и iOS?

Аннотация. Работа посвящена исследованию кросплатформенного фреймворка для разработки мобильных приложений Flutter от Google. Проведено практическое испытание технологии с целью установить, достаточно ли она удобная и надежная. В статье представлены выводы по результатам проведенных исследований. Приведено сравнение механизмов работы Flutter и других популярных мобильных кроссплатформенных фреймворков. Рассмотрены основные особенности и отличия технологии. Приведены рекомендации по ее использованию.

Ключевые слова: кроссплатформенные решения; мобильные приложения; нативная разработка; язык программирования Dart.

УДК 681.142:681 DOI: 10.26565/2519-2310-2019-2-04

МЕТОД КОНТРОЛЮ ДАНИХ, ПРЕДСТАВЛЕНИХ КОДОМ НЕПОЗИЦІЙНОЇ СИСТЕМИ ЧИСЛЕННЯ КЛАСІВ ЗАЛИШКІВ

Андрій Д'яченко, Ірина Локоткова, Олеся Решетняк, Михайло Зуб, Костянтин Мисливцев

Харківський національний університет імені В.Н. Каразіна, майдан Свободи 4, Харків, 61022, Україна
0andrey.090220@gmail.com, lokotosyk@ukr.net, lesyandr13@gmail.com, mishazub007@gmail.com, kostyavtx@gmail.com

Рецензент: Олександр Оксіюк, д.т.н., проф., Київський національний університет імені Т. Шевченка, вул. М. Ломоносова 81, Київ, 03189, Україна.
o.oksiuk@gmail.com

Надійшло: Жовтень 2019.

Анотація: *Пропонується новий метод моніторингу даних, представлених у непозиційній системі класів залишків. Для коду в системі класів залишків тестові бази включаються в загальну структуру коду даних, що містить набір баз інформації. У цьому випадку баланси, які представляють операції з інформаційних та контрольних підстав одночасно та незалежно беруть участь у процесі обробки інформації. За результатами обробки інформації можна відслідковувати або поетапно, або наприкінці всіх обчислень, оскільки помилка, яка сталася в будь-якому залишку, не застосовується (не "помножується") на залишки, що залишилися. Запропонований метод контролю, заснований на принципі порівняння, в подальшому створює передумови для розробки ефективних методів діагностики і корекції помилок в класі залишків. Недоліком запропонованого методу є дещо низька оперативність контролю. Дана обставина обумовлює необхідність підвищення оперативності контролю системи обробки даних в класі залишків за рахунок зменшення часу виконання перерахованих вище операцій шляхом розробки і використання, наприклад, методів і засобів реалізації позиційних ознак непозиційних кодів у класі залишків.*

Ключові слова: *метод контролю даних, клас залишків; непозиційна система.*

1 Вступ

Відомо, що використання властивостей непозиційної системи числення класу залишків (КЗ) забезпечує високу призначену для користувача продуктивність реалізації в системі обробки даних (СОД) обчислювальних алгоритмів, що складаються з арифметичних операцій [1-7]. Найбільша ефективність від застосування КЗ досягається в разі, коли реалізовані алгоритми складаються з сукупності таких арифметичних операцій, як додавання і віднімання множення [8-14]. Необхідність забезпечення відмовостійкого функціонування СОД вимагає розробки і впровадження в КЗ нових методів контролю і корекції помилок інформації, відмінних від методів, використовуваних в звичайних двійкових позиційних системах числення (ПСЧ) [15-21].

Відзначимо, що, по-перше, всі методи контролю та корекції даних в КЗ ґрунтуються на специфіці реалізації позиційних операцій в даній системі числення, що вимагає знання величини (можливо знаку) числа, представленого даною непозиційною кодовою структурою. По-друге, методи контролю в КЗ є подальшим розвитком контролю за модулем в ПСЧ (*арифметичні AN-коди*) [1-3, 10-14]. Дійсно, з точки зору інформаційного резервування коди є подальшим вдосконаленням відомих арифметичних багатозалишкових кодів.

Відомо, що багатозалишковий код представляється у вигляді:

$$A_k^{'} = (A_k, A_k (\mathrm{mod}\, m_1), A_k (\mathrm{mod}\, m_2), ..., A_k (\mathrm{mod}\, m_i), ...,$$
$$..., A_k (\mathrm{mod}\, m_{n-1}), A_k (\mathrm{mod}\, m_n),$$

тобто $A_k^{'} = (A_k, a_1, a_2, \ldots, a_n)$, де $a_i = A_k - [A_k / m_i] m_i$.

У цьому випадку, для

$$\prod_{i=1}^{n} m_i \geq A_k,$$

сукупність залишків $\{a_i\}$ однозначно визначає операнд $A'_к$ і чисельне значення $A_к$ стає взагалі не потрібне. Багатозалишковий код набирає вигляду непозиційного коду КЗ

$$A_к^{'} = (a_1, a_2, \ldots, a_n),$$

що дозволяє реалізувати модульні операції по окремим незалежним трактам, оперуючи тільки із залишками $\{a_i\}$. Таке кодування чисел дозволяє побудувати СОД, в якому обробка всіх залишків a_i числа проводиться паралельно в часі [1, 2]. В цьому випадку узагальнена структурна схема СОД в КЗ являє собою набір окремих мікро-ЕОМ, що функціонують незалежно один від одного і паралельно в часі, причому кожна по власному певному модулю m_i [12-14].

2 Основна частина

Процес корекції (виявлення і виправлення) помилок в інформаційній кодової структурі Ã даних складається з наступних основних етапів:

- контроль даних (*процес виявлення факту наявності помилки в числі A*);
- діагностика (*локалізація місця помилок із заданою глибиною діагностування*);
- виправлення помилок в кодовій структурі даних (*відновлення спотворених залишків*

$$\tilde{a}_j (j = \overline{1, n})$$

числа Ã та отримання правильного числа A).

Число

$$A = (a_1, a_2, \ldots, a_{i-1}, a_i, a_{i+1}, \ldots, a_n)$$

в КЗ представляється сукупністю

$$\{a_i\} (i = \overline{1, n})$$

залишків

$$a_i \equiv A (\mathrm{mod}\, m_i)$$

по вибраній системі інформаційних основ (*модулів*) $\{m_i\}$ в робочому (*інформаційному*) числовому інтервалі [0, M), де

$M = \prod_{i=1}^{n} m_i$ – загальна кількість інформаційних кодових слів.

При цьому НСД

$$(m_i, m_j) = 1,$$

при

$$i, j = \overline{1, n}\ (i \neq j).$$

Для того, щоб код КЗ володів коректуючими властивостями необхідно, щоб він містив певну інформаційну надмірність. При цьому, по-перше, необхідно визначити (виявити) і, по можливості, кількісно оцінити наявну (природну) в первісній інформаційній кодовій структурі надмірність. По-друге, при необхідності забезпечення даних додатковими коректуючими здібностями, ввести додаткову (штучну) інформаційну надмірність (застосувати методи інформаційного резервування) за рахунок введення додаткових (контрольних) підстав $\{mk\}$ КЗ. Без втрати спільності міркувань вважатимемо, що до n інформаційним підставою КЗ додано одне

$$\mathrm{m_k} = \mathrm{m_{n+1}}$$

контрольне підставу взаємно просте з будь-яким з наявних інформаційних підстав. У цьому випадку число

$$A = (a_1, a_2, \ldots, a_n, a_{n+1})$$

в КЗ представляється за допомогою сукупності

$$\{m_j\}\ (j = \overline{1, n+1})$$

основ у повному числовому [0, M_0) інтервалі, де $M_0 = M \cdot m_{n+1}$ – загальна кількість кодових слів для даного КЗ.

Відомо [1-3], що для непозиційних кодових структур в КЗ мінімальна кодова відстань визначається виразом

$$d_{\min} = K + 1,$$

тобто воно залежить як від числа до контрольних підстав, так і від величини кожного з них. Якщо для контрольних основ m_{z_t} виконується умова

$$\prod_{t=1}^{k} m_{z_t} \le m_k,$$

тоді введення в систему підстав КЗ однієї контрольної

$$m_k = m_{n+1}$$

основи еквівалентно наявності К контрольних основ. З урахуванням того, що всі числа, які беруть участі в обробці даних (передача та обробка інформації), а також результат операції даними знаходиться в інтервалі [0, M), то очевидно, що якщо в результаті обробки даних отримано остаточний результат A ≥ M, то це означає, що отримане число Ã спотворене (неправильне).

Таким чином, якщо в результаті обробки даних визначено, що Ã ≥ M, то робиться висновок, що число Ã неправильне. На цьому принципі (принципі порівняння) ґрунтуються всі методи порівняння даних в КЗ. Якщо A <M, то робиться висновок, що число А правильне, а якщо A ≥ M, то число Ã неправильне. При цьому передбачаються тільки одноразові (в одному із залишків $\{a_i\}$ числа А) помилки, або пачка помилок довжиною не більше

$$k = [\log_2 (m_i - 1)] + 1$$

двійкових розрядів.

Для розгляду методу контролю (*метод прямого порівняння*) даних в КЗ на основі зазначеного вище принципу, який використовується так само при розробці методів діагностики і корекції помилок, скористаємося результатами докази відомого [1] наукового затвердження.

Твердження. Нехай інформаційна основа

$$\{m_i\} \ (i = \overline{1, n})$$

КЗ з однією контрольною

$$m_k = m_{n+1}$$

основою впорядковані ($m_i < m_{i+1}$), і нехай результат

$$A = (a_1, a_2, ..., a_{i-1}, a_i, a_{i+1}, ..., a_n, a_{n+1})$$

виконання операції є правильним числом, тобто

$$A < M \ (M = \prod_{i=1}^{n} m_i).$$

Тоді стверджується, що число

$$\tilde{A} = (a_1, a_2, ..., a_{i-1}, \tilde{a}_i, a_{i+1}, ..., a_n, a_{n+1}),$$

в якому спотворений один залишок $\tilde{a}_i \neq a_i$ за основою m_i, є неправильним, тобто $\tilde{A} \ge M$.

Продемонструємо це.

На основі принципу порівняння правильність числа А визначається виходячи зі співвідношення.

$$A < M = M_0 / m_{n+1} \ (M_0 = \prod_{i=1}^{n+1} m_i)$$

З іншого боку, очевидно виконання умови $M_0 / m_i > M_0 / m_{n+1}$ для упорядкованого КЗ при

$$i = \overline{1, n}.$$

В цьому випадку виконується така нерівність $A < M_0 / m_i$.

Відзначимо, що залишок

$$a_i \equiv A(\mathrm{mod}\, m_i)$$

числа А по модулю m_i може набувати значень

$$a_i = \overline{0, m_i - 1}.$$

У відповідності до розділу твердження, що

$$\tilde{a}_i \neq a_i,$$

і враховуючи, що інші значення залишків

$$a_j (j = \overline{1, n+1} \text{ та } i \neq j)$$

неправильного Ã числа залишаються без змін, то в інтервалі [0, M_0/m_i) не можуть одночасно знаходиться обидва числа А і Ã. Тоді, так як число А < M_0/m_{n+1} правильне (*знаходиться в робочому [0, M]інтервалі*), то число Ã знаходиться поза інтервалом [0, M_0/m_i), і тим більше знаходиться поза інтервалом [0, M).

У цьому випадку число Ã, для якого виконується умова $\tilde{A} \geq M$, є неправильним. Таким чином показано, що число

$$A = (a_1, a_2, \ldots, a_{i-1}, \tilde{a}_i, a_{i+1}, \ldots, a_n, a_{n+1})$$

є спотвореним (Рис 1, 2).

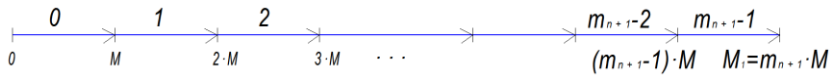


Рис. 1 - Числові інтервали для $m_k = m_{n+1}$

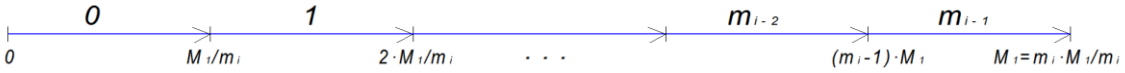


Рис. 2 - Числові інтервалі для довільного модуля *mi*

3 Аналіз методу контролю даних

Розглянемо метод контролю даних в КЗ, заснований на результатах і висновках розглянутого наукового затвердження.

В основі контролю лежить базова операція - порівняння результату А операції з числом

$$M = M_0/m_{n+1}.$$

Для порівняння чисел

$$A = (a_1, a_2, \ldots, a_n, a_{n+1})$$

та М необхідно перевести значення А в позиційну двійкову систему числення (ПСЧ). Для цього можна використовувати ортогональні базиси

$$B_i (i = \overline{1, n+1}),$$

яке представляються у вигляді [1-3]:

$$\begin{cases} B_1 = (1,0,\ldots,0,\ldots,0,0), \\ B_2 = (0,1,\ldots,0,\ldots,0,0), \\ B_i = (0,0,\ldots,1,\ldots,0,0), \\ B_n = (0,0,\ldots,0,\ldots,1,0), \\ B_{n+1} = (0,0,\ldots,0,\ldots,0,1). \end{cases} \quad (1)$$

Ортогональні базиси B_i визначаються для кожного КЗ відповідно до виразу

$$B_i = \overline{m}_i \cdot M_0 / m_i \equiv 1(\mathrm{mod}\, m_i). \quad (2)$$

Значення ваги $\overline{m}_i$ ортогонального базису B_i визначається як одне з рішень системи порівнянь:

$$\begin{cases} \overline{m}_i = 1, \quad 1 \cdot M_i \equiv \rho_1 (\mathrm{mod}\ m_i), \\ \overline{m}_i = 2, \quad 2 \cdot M_i \equiv \rho_2 (\mathrm{mod}\ m_i), \\ \qquad \ldots \\ \overline{m}_i = m_i - 2, \quad (m_i - 2) \cdot M_i \equiv \rho_{m-2} (\mathrm{mod}\ m_i), \\ \overline{m}_i = m_i - 1, \quad (m_i - 1) \cdot M_i \equiv \rho_{m-1} (\mathrm{mod}\ m_i). \end{cases} \tag{3}$$

Значення $\overline{m}_i$, для якого виконується умова (2), визначається шляхом підстановки можливих значень

$$\overline{m}_i = \overline{1, m_i - 1}$$

методом простого перебору. Значення А в ПСЧ визначається у співвідношенні з формулою

$$A_{ПСЧ} = (\sum_{i=1}^{n+1} a_i \cdot B_i) \bmod M_0 .$$

4. Приклади конкретної реалізації методу контролю даних в КЗ

Розглянемо приклади реалізації методу контролю даних для упорядкованого КЗ, заданого інформаційними $m_1 = 3, m_2 = 4, m_3 = 5, m_4 = 7$ і контрольним $m_k = m_5 = 11$ основами (рис. 3).

Даний КЗ забезпечує інформаційний інтервал [0, M) для однобайтової (l=1) СОД, де

$$M = \prod_{i=1}^{4} m_i = 420 .$$

Повний числовий інтервал уявлення числі в КЗ визначається як [0, M_0) (див. рис.3), де

$$M_0 = M \cdot m_{n+1} = 4620.$$

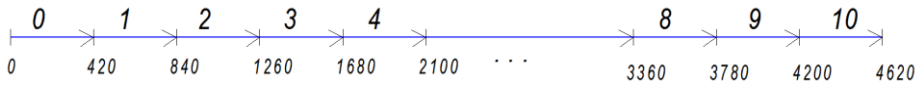


Рис. 3 - Числові інтервали для $l = 1$ ($m_k = m_{n+1} = 11$)

У таблицях 1÷5 представлені процедури визначення значень $\overline{m}_i (i = \overline{1,5})$, а в таблиці 6 розраховані значення ортогональних базисів $B_i (i = \overline{1,5})$.

Табл. 1 – Процедура визначення $\overline{m}_1$

$\overline{m}_1 = 3, M_1 = 4 \cdot 5 \cdot 7 \cdot 11 = 1540$	
$\overline{m}_1 = 1$	$\overline{m}_1 \cdot M_1 = 1540 \equiv 1 (\mathrm{mod}\, m_1)$
$\overline{m}_2 = 2$	$\overline{m}_1 \cdot M_1 = 3080 \equiv 2 (\mathrm{mod}\, m_1)$
$\overline{m}_1 = 1, B_1 = (1,0,0,0,0) = 1540$	

Табл. 2 – Процедура визначення $\overline{m}_2$

$m_2 = 4, M_2 = 3 \cdot 5 \cdot 7 \cdot 11 = 1155$	
$\overline{m}_2 = 1$	$1 \cdot M_2 = 1155 \equiv 3 (\mathrm{mod}\, m_2)$
$\overline{m}_2 = 2$	$2 \cdot M_2 = 2310 \equiv 2 (\mathrm{mod}\, m_2)$
$\overline{m}_2 = 3$	$3 \cdot M_2 = 3465 \equiv 1 (\mathrm{mod}\, m_2)$
$\overline{m}_2 = 3$	$B_2 = (0,1,0,0,0) = 3465$

Табл. 3 – Процедура визначення $\overline{m}_3$

$m_3 = 5, M_3 = 3 \cdot 4 \cdot 7 \cdot 11 = 924$	
$\overline{m}_3 = 1$	$1 \cdot M_3 = 924 \equiv 4 (\mathrm{mod}\, m_3)$
$\overline{m}_3 = 2$	$2 \cdot M_3 = 1848 \equiv 3 (\mathrm{mod}\, m_3)$
$\overline{m}_3 = 3$	$3 \cdot M_3 = 2772 \equiv 2 (\mathrm{mod}\, m_3)$
$\overline{m}_3 = 4$	$4 \cdot M_2 = 3696 \equiv 1 (\mathrm{mod}\, m_3)$
$\overline{m}_3 = 4,\ B_3 = (0,0,1,0,0) = 3696$	

Табл. 4 – Процедура визначення $\overline{m}_4$

$m_4 = 7, M_4 = 3 \cdot 4 \cdot 5 \cdot 11 = 660$	
$\overline{m}_4 = 1$	$1 \cdot M_4 = 660 \equiv 2 (\mathrm{mod}\, m_4)$
$\overline{m}_4 = 2$	$2 \cdot M_4 = 1320 \equiv 4 (\mathrm{mod}\, m_4)$
$\overline{m}_4 = 3$	$3 \cdot M_4 = 1980 \equiv 6 (\mathrm{mod}\, m_4)$
$\overline{m}_4 = 4$	$4 \cdot M_4 = 2640 \equiv 1 (\mathrm{mod}\, m_4)$
$\overline{m}_4 = 5$	$5 \cdot M_4 = 3300 \equiv 3 (\mathrm{mod}\, m_4)$
$\overline{m}_4 = 6$	$6 \cdot M_4 = 3960 \equiv 5 (\mathrm{mod}\, m_4)$
$\overline{m}_4 = 4,\ B_4 = (0,0,0,1,0) = 2640$	

Табл. 5 – Процедура визначення $\overline{m}_5$

$m_5 = 11, M_5 = 3 \cdot 4 \cdot 5 \cdot 7 = 420$	
$\overline{m}_5 = 1$	$1 \cdot M_5 = 420 \equiv 2 (\mathrm{mod}\, m_5)$
$\overline{m}_5 = 2$	$2 \cdot M_5 = 840 \equiv 4 (\mathrm{mod}\, m_5)$
$\overline{m}_5 = 3$	$3 \cdot M_5 = 1260 \equiv 6 (\mathrm{mod}\, m_5)$
$\overline{m}_5 = 4$	$4 \cdot M_5 = 1680 \equiv 8 (\mathrm{mod}\, m_5)$
$\overline{m}_5 = 5$	$5 \cdot M_5 = 2100 \equiv 10 (\mathrm{mod}\, m_5)$
$\overline{m}_5 = 6$	$6 \cdot M_5 = 2520 \equiv 1 (\mathrm{mod}\, m_5)$
$\overline{m}_5 = 7$	$7 \cdot M_5 = 2940 \equiv 3 (\mathrm{mod}\, m_5)$
$\overline{m}_5 = 8$	$8 \cdot M_5 = 3360 \equiv 5 (\mathrm{mod}\, m_5)$
$\overline{m}_5 = 9$	$9 \cdot M_5 = 3780 \equiv 7 (\mathrm{mod}\, m_5)$
$\overline{m}_5 = 10$	$10 \cdot M_5 = 4200 \equiv 9 (\mathrm{mod}\, m_5)$
$\overline{m}_5 = 6,\ B_5 = (0,0,0,0,1) = 2520$	

Табл. 6 – Ортогональні базиси B_i КЗ

$B_1 = (1,0,0,0,0) = 1540$
$B_2 = (0,1,0,0,0) = 3465$
$B_3 = (0,0,1,0,0) = 3696$
$B_4 = (0,0,0,1,0) = 2640$
$B_5 = (0,0,0,0,1) = 2520$

Нехай задано правильне

$$A_{400} = (1,0,0,1,4)$$

число в КЗ.

Приклад 1. Визначити правильність чисел $\tilde{A} = (1,0,0,1,4)$, спотвореного по підставі

$$\mathrm{m}_1 = 3\ (\tilde{a}_1 = 0).$$

Переводимо число Ã в ПСЧ та порівнюємо його з M = 420.

$$\tilde{A} = (\sum_{i=1}^{n+1} a_i \cdot B_i) \bmod M_1 = (\sum_{i=1}^{5} a_i \cdot B_i) \bmod 4620 =$$

$$= (1540 \cdot 0 + 3465 \cdot 0 + 3696 \cdot 0 + 2640 \cdot 1 + 2520 \cdot 4) \ \bmod(4620) = 3480 > 420 .$$

$$\bmod 4620 = 12720$$

Таким чином операнд Ã містить помилку в одному з п'яти залишків.

Приклад 2. Нехай число $A_{400} = (1,0,0,1,4)$ неспотворене.

В цьому випадку отримаємо

$$A = (1540 \cdot 1 + 3465 \cdot 0 + 3696 \cdot 0 + 2640 \cdot 1 + 2520 \cdot 4)$$

$$\bmod 4620 = 14260(\bmod 4620) = 400 < 420.$$

Т.ч. число А є правильне.

5 Висновки

1. Аналіз властивостей непозиційних кодових структур показав, що коди в КЗ є арифметичними кодами (*деякий аналог цих кодів в ПСС - арифметичні AN-коди*), придатні до використання, як для передачі, так і для обробки інформації. Для кодів КЗ контрольні основи включені в загальну кодову структуру даних, що містять сукупність інформаційних підстав. В цьому випадку залишки, якими представляються операції в КЗ з інформаційних і контрольних підстав, одночасно і незалежно беруть участь в процесі обробки інформації. Результат обробки інформації, представлений кодом КЗ, може контролюватися або поетапно, або по закінченню всіх обчислень, так як помилка, що виникла в будь-якому залишку a_i числа

$$A = (a_1, a_2, \ldots, a_{i-1}, a_i, a_{i+1}, \ldots, a_n, a_{n+1}),$$

не поширюється (не "розмножується") в інші залишки

$$a_j \ (j = \overline{1, n+1} \ \text{ та } \ i \neq j)$$

числа А (*у сусідні тракти обробки інформації СОД*). Важливе значення має факт визначення кількості і величин (за кількістю реалізованих операцій) кожного з цих етапів. Наприклад, величина кожного етапу обробки інформації (етапу обчислень) може бути визначена або по обумовленому замкненим циклом обробки алгоритму, або відповідно до можливого значення ймовірності виникнення одноразової (в одному залишку числа в КЗ) помилки.

2. Для довільної впорядкованої

$$(m_i < m_{i+1})$$

системи підстав КЗ, з одним контрольним *mk* підставою спотворення одного довільного залишку aj по модулю m_j однозначно перетворює правильне (*неспотворене*) число А в неправильне Ã. В цьому випадку система контролю в КЗ достовірно визначає факт спотворення числа А.

Коригувальні здібності перешкодостійкого коду в КЗ залежить від кількості та величини контрольних $\{m_k\}$ основ. Якщо для деякої кількості r інформаційних підстав даного КЗ виконується умова

$$\prod_{i=1}^{r} m_{z_i} \leq m_k \ (m_{z_i} \in M),$$

то спотворення в одночасно декількох або навіть у всіх цих r залишках не робить правильне число А в неправильне. При цьому вважається, що СОД функціонує безвідмовно. Однак при цьому система контролю СОД не визначає номера відмовили трактів обробки даних.

3. Запропонований метод контролю, заснований на принципі порівняння, в подальшому створює відповідні передумови для розробки ефективних методів діагностики і корекції помилок в КЗ. Недоліком запропонованого методу є дещо низька оперативність контролю. Цей недолік обумовлений значними тимчасовими витратами на операції переказу числа А з КЗ в ПСЧ та порівняння чисел А і М в ПСЧ. Крім цього, якщо контролю піддається проміжний

результат обчислень, то можливо додатково буде потрібно додаткова операція перекладу числа А з ПСЧ в КЗ. Дана обставина обумовлює необхідність підвищення оперативності контролю СОД в КЗ за рахунок зменшення часу виконання перерахованих вище операцій шляхом розробки і використання, наприклад, методів і засобів реалізації позиційних ознак непозиційних коду КЗ.

4. Перспективним напрямком подальших досліджень є розробка практичних рекомендацій щодо адаптації запропонованого методу до специфіки завдань криптографії [22-30], цифрової обробки сигналів [20, 21, 31-33], та вирішення комплексу інших [34-42] обчислювальних задач.

Посилання

[1] Akushsky I. Ya.,Yuditsky DI. Machine arithmetic in residual classes. Moscow: Soviet radio, 1968. 440 p. (In Russian)

[2] Torgashov V. A. The system of residual classes and reliability of digital computersю Moscow: Soviet radio, 1973. 118 p. (In Russian)

[3] Improved Method of Determining the Alternative Set of Numbers in Residue Number System/ Krasnobayev V., Kuznetsov A., Koshman S., Moroz S. *Recent Developments in Data Science and Intelligent Analysis of Information*. *ICDSIAI 2018*. Advances in Intelligent Systems and Computing. Springer, Cham, 2018. Vol. 836. P. 319–328.

[4] Barsi F., Maestrini P. Error Correcting Properties of Redundant Residue Number Systems. *IEEE Transactions on Computers*. 1973. Vol. C-22, № 3, P. 307–315.

[5] Krasnobayev V.A., Yanko A.S., Koshman S.A.A Method for arithmetic comparison of data represented in a residue number system. *Cybernetics and Systems Analysis*. 2016.Vol. 52, Issue 1.P. 145–150.

[6] Harman G., Shparlinski I. E. Products of Small Integers in Residue Classes and Additive Properties of Fermat Quotients. *International Mathematics Research Notices*. 2016. № 5. P. 1424–1446.

[7] Methods for comparing numbers in non-positional notation of residual classes/ V. Krasnobayev, A. Kuznetsov, M. Zub, K. Kuznetsova. *Proceedings of the Second International Workshop on Computer Modeling and Intelligent Systems (CMIS-2019)*. Zaporizhzhia, 2019. P. 581–595.

[8] Popov D. I., Gapochkin A. V. Development of Algorithm for Control and Correction of Errors of Digital Signals, Represented in System of Residual Classes. *2018 International Russian Automation Conference (RusAutoCon)*. Sochi, 2018. P. 1–3.

[9] Advanced method of factorization of multi-bit numbers based on Fermat's theorem in the system of residual classes / M. Karpinski, S. Ivasiev, I. Yakymenko, M. Kasianchuk and T. Gancarczyk. *16th International Conference on Control, Automation and Systems (ICCAS)*. Gyeongju, 2016. P.1484–1486.

[10] A Method for Increasing the Reliability of Verification of Data Represented in a Residue Number System/ V.A. Krasnobayev, S.A. Koshman, M.A. Mavrina. *Cybernetics and Systems Analysis*. 2014. Vol. 50, Issue 6. P. 969–976.

[11] Irregular repeat accumulate low-density parity-check codes based on residue class pair/ K. Tao, L. Peng, K. Liang and B. Zhuo. *IEEE 9th International Conference on Communication Software and Networks (ICCSN)*. Guangzhou, 2017. P.127–131.

[12] Method of data control in the residue classes/ V. Krasnobayev, A. Kuznetsov, A. Kononchenko, T. Kuznetsova. *Proceedings of the Second International Workshop on Computer Modeling and Intelligent Systems (CMIS-2019)*. Zaporizhzhia, 2019. P.241–252.

[13] Method of Error Control of the Information Presented in the Modular Number System/ V. Krasnobayev, S. Koshman, A. Yanko and A. Martynenko. *International Scientific-Practical Conference Problems of Infocommunications. Science and Technology (PIC S&T)*. Kharkiv, 2018. P. 39–42.

[14] Krasnobayev V. A. Method for realization of transformations in public-key cryptography. *Telecommunications and Radio Engineering*. 2007.Vol. 66, Issue 17. P. 1559–1572.

[15] Gorbenko I.D., Zamula A.A. Cryptographic signals: requirements, methods of synthesis, properties, application in telecommunication systems. *Telecommunications and Radio Engineering*. 2017. Vol. 76, Issue 12. P. 1079–1100.

[16] Kuznetsov A., Serhiienko R., Prokopovych-Tkachenko D. Construction of cascade codes in the frequency domain. *4th International Scientific-Practical Conference Problems of Infocommunications. Science and Technology (PIC S&T)*. Kharkov, 2017. P.131–136.

[17] Stasev Yu.V., Kuznetsov A.A., Nosik A.M. Formation of pseudorandom sequences with improved autocorrelation properties. *Cybernetics and Systems Analysis*. 2007. Vol. 43, Issue 1. P. 1–11.

[18] Soft decoding based on ordered subsets of verification equations of turbo-productive codes/ A. Kuznetsov, A. Kiian, K. Kuznetsova, et al. *Proceedings of the Second International Workshop on Computer Modeling and Intelligent Systems (CMIS-2019)*. Zaporizhzhia, 2019. P. 873–884.

[19] Analysis and Studying of the Properties of Algebraic Geometric Codes/ A. A. Kuznetsov, I. P. Kolovanova, D. I. Prokopovych-Tkachenko, T. Y. Kuznetsova. *Telecommunications and Radio Engineering*. 2019. Vol. 78, Issue 5. P. 393–417.

[20] Puschel M., Moura J. M. F. Algebraic Signal Processing Theory: Foundation and 1-D Time. *IEEE Transactions on Signal Processing*. 2008. Vol. 56, №8. P. 3572–3585.

[21] Discrete Signals with Multi-Level Correlation Function/ O.Karpenko, A.Kuznetsov, V.Sai, Yu.Stasev. *Telecommunications and Radio Engineering*. 2012. Vol.71, Issue 1. P. 91–98.

[22] Stasev Yu.V., Kuznetsov A.A. Asymmetric code-theoretical schemes constructed with the use of algebraic geometric codes. *Kibernetika i Sistemnyi Analiz*. 2005. № 3. P. 47–57.

[23] Agarwala A., Saravanan R. A Public Key Cryptosystem based on number theory. *International Conference on Recent Advances in Computing and Software Systems*. Chennai, 2012. P. 238–241.

[24] Code-based public-key cryptosystems for the post-quantum period/ A. Kuznetsov, I. Svatovskij, N. Kiyan and A. Pushkar'ov. *4th International Scientific-Practical Conference Problems of Infocommunications. Science and Technology (PIC S&T)*. Kharkov, 2017. P. 125–130.

[25] Wang B., Liu L. A flexible and energy-efficient reconfigurable architecture for symmetric cipher processing. *2015 IEEE International Symposium on Circuits and Systems (ISCAS)*. Lisbon, 2015. P.1182–1185.

[26] Analysis of block symmetric algorithms from international standard of lightweight cryptography ISO/IEC 29192-2/ A. Kuznetsov, Y. Gorbenko, A. Andrushkevych and I. Belozersev. *4th International Scientific-Practical Conference Problems of Infocommunications. Science and Technology (PIC S&T)*. Kharkov, 2017. P. 203–206.

[27] The research of modern stream ciphers/ I. Gorbenko, A. Kuznetsov, M. Lutsenko and D. Ivanenko. *4th International Scientific-Practical Conference Problems of Infocommunications. Science and Technology (PIC S&T)*. Kharkov, 2017. P. 207–210.

[28] Lightweight robust cryptographic combiner for mobile devices: Crypto roulette/ M. E. Pamukov, V. Poulkov, A. Mihovska, N. R. Prasad and R. Prasad. *IEEE 19th International Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD)*. Athens, 2014. P. 188–192.

[29] Periodic characteristics of output feedback encryption mode/ A. Kuznetsov, I. Kolovanova and T. Kuznetsova. *4th International Scientific-Practical Conference Problems of Infocommunications. Science and Technology (PIC S&T)*. Kharkov, 2017. P. 193–198.

[30] Thangavel M., Varalakshmi P. A novel public key cryptosystem based on Merkle-Hellman Knapsack Cryptosystem. *Eighth International Conference on Advanced Computing (ICoAC)*. Chennai, 2017. P. 117–122.

[31] Gorbenko I., Nariezhnii O., Kudryashov I. Construction method and features of one class of cryptographic discrete signals. *4th International Scientific-Practical Conference Problems of Infocommunications. Science and Technology (PIC S&T)*. Kharkov, 2017. P.156–160.

[32] Ahmed N., Natarajan T., Rao K. R. Discrete cosine transform. *IEEE Trans. Comput.* 1974. Vol. C-23, № 1, P. 90–93.

[33] Naumenko N.I., Stasev Yu.V., Kuznetsov A.A. Methods of synthesis of signals with prescribed properties. *Cybernetics and Systems Analysis*. 2007. Vol. 43, Issue 3. P. 321–326.

[34] Accelerated Flexible Processor Architecture for Crypto Information/ Z. Dai, X. Yu, J. Su and X. Chen. *2nd International Conference on Pervasive Computing and Applications*. Birmingham, 2007. P. 399–403.

[35] Strumok keystream generator/ I. Gorbenko, O. Kuznetsov, Y. Gorbenko, A. Alekseychuk and V. Tymchenko. *IEEE 9th International Conference on Dependable Systems, Services and Technologies (DESSERT)*. Kyiv, 2018. P. 294–299.

[36] Runovski, K., Schmeisser, H. On the convergence of fourier means and interpolation means. *Journal of Computational Analysis and Applications*. 2004. № 6(3). P. 211–227.

[37] Gnatyuk, V. A. Mechanism of laser damage of transparent semiconductors. *Physica B: Condensed Matter*. 2001. P. 308-310; P. 935–938.

[38] Tkach, B. P., Urmancheva, L. B. Numerical-analytic method for finding solutions of systems with distributed parameters and integral condition. *Nonlinear Oscillations*. 2009. №12(1). P.113–122.

[39] Controlled markov fields with finite state space on graphs/ Chornei, R., Hans Daduna, V. M., Knopov, P. *Stochastic Models.* 2005. Issue 21(4). P. 847–874.

[40] Development of an Antinoise Method of Data Sharing Based on the Application of a Two-Step-Up System of Residual Classes/ Y. N. Kocherov, D. V. Samoylenko and A. I. Koldaev. *International Multi-Conference on Industrial Engineering and Modern Technologies (FarEastCon).* Vladivostok, 2018. P. 1–5.

[41] Fan C. , Ge G. A Unified Approach to Whiteman's and Ding-Helleseth's Generalized Cyclotomy Over Residue Class Rings. *IEEE Transactions on Information Theory*. 2014.Vol. 60, № 2. P.1326–1336.

[42] Rabin's modified method of encryption using various forms of system of residual classes/ M. Kasianchuk, I. Yakymenko, I. Pazdriy, A. Melnyk and S. Ivasiev. *14th International Conference The Experience of Designing and Application of CAD Systems in Microelectronics (CADSM)*. Lviv, 2017. P. 222–224.

Reviewer: Oleksandr Oksiiuk, Doctor of Sciences (Engineering), Full Professor, Taras Shevchenko National University of Kiev 81 Lomonosova St., Kyiv, 03189, Ukraine. E-mail: o.oksiuk@gmail.com

Received on October 2019.

Authors:

Andrey Dyachenko, student, V.N. Karazin Kharkiv National University, Kharkiv, Ukraine.
E-mail: andrey.090220@gmail.com

Irina Lokotkova, student, V.N. Karazin Kharkiv National University, Kharkiv, Ukraine.
E-mail: lokotosyk@ukr.net

Olesya Reshetnyak, student, V.N. Karazin Kharkiv National University, Kharkiv, Ukraine.
E-mail: lesyandr13@gmail.com

Mikhail Zub, student, V.N. Karazin Kharkiv National University, Kharkiv, Ukraine.
E-mail: mishazub007@gmail.com

Konstantin Myslyvtsev, student, V.N. Karazin Kharkiv National University, Kharkiv, Ukraine.
E-mail: kostyavtx@gmail.com

Data control method, which presented by code of non-positioning system of deduction class calculation.

Abstract. The paper proposes a new method of monitoring data presented in non-positional residue class system. For code in residue class system, test bases are included in the general code structure of data containing a set of information bases. In this case, the balances that represent operations for informational and control grounds simultaneously and independently participate in the process of

information processing. The result of the information processing can be monitored either step by step or at the end of all calculations, since the error that occurred in any residue, does not apply (does not “multiply”) to the remaining residues. The control method proposed based on the principle of comparison, further creates prerequisites for developing effective methods for diagnosing and correcting errors in deduction class. The disadvantage of the proposed method is the relatively low control efficiency. This circumstance makes it necessary to increase the efficiency of data processing system control in deduction class by reducing the execution time of the above operations by developing and using, for example, methods and means for implementing the positional features of the non-positional deduction class code.

Keywords: Data Control Method; Residues Class; Non-Positioning System.

Рецензент: Александр Оксиюк, д.т.н., проф., Киевский национальный университет имени Т. Шевченко, ул. Ломоносова 81, Киев, 03189 Украина. E-mail: o.oksiuk@gmail.com

Поступила: Октябрь 2019.

Авторы:

Андрей Дьяченко, студент, Харьковский национальный университет им. В.Н. Каразина.
E-mail: andrey.090220@gmail.com

Ирина Локоткова, студент, Харьковский национальный университет им. В.Н. Каразина.
E-mail: lokotosyk@ukr.net

Олеся Решетняк, студент, Харьковский национальный университет им. В.Н. Каразина.
E-mail: lesyandr13@gmail.com

Михаил Зуб, студент, Харьковский национальный университет им. В.Н. Каразина.
E-mail: mishazub007@gmail.com

Константин Мисливцев, студент, Харьковский национальный университет им. В.Н. Каразина.
E-mail: kostyavtx@gmail.com

Метод контроля данных, представленных кодом непозиционной системы счисления класса вычетов.

Аннотация. В статье предлагается новый метод мониторинга данных, представленных в непозиционной системе классов вычетов. Для кода в системе классов вычетов тестовые базы включаются в общую структуру кода данных, содержащий набор баз информации. В этом случае балансы, которые представляют операции информационных и контрольных оснований одновременно и независимо участвуют в процессе обработки информации. По результатам обработки информации можно отслеживать или поэтапно, или в конце всех вычислений, поскольку ошибка, которая произошла в любом из вычетов, не применяется (не «умножается») на остатки, которые остались. Предложенный метод контроля, основанный на принципе сравнения, в дальнейшем создает предпосылки для разработки эффективных методов диагностики и коррекции ошибок в классе вычетов. Недостатком предложенного метода является относительно низкая оперативность контроля. Данное обстоятельство обусловливает необходимость повышения оперативности контроля системы обработки данных в классе вычетов за счет уменьшения времени выполнения вышеперечисленных операций путем разработки и использования, например, методов и средств реализации позиционных признаков непозиционных кодов в классе вычетов.

Ключевые слова: метод контроля данных; класс вычетов; непозиционная система.

УДК 004.057.4 DOI: 10.26565/2519-2310-2019-2-05

ВИМОГИ ДО СТВОРЕННЯ ІНСТРУМЕНТАЛЬНИХ ЗАСОБІВ ДЛЯ ЕМУЛЯЦІЇ ПРОЦЕСІВ ІНТЕГРАЦІЇ ПРОГРАМНИХ СИСТЕМ ІЗ ВИКОРИСТАННЯМ HTTP/HTTPS ПРОТОКОЛІВ

Костянтин Д'яченко, Дмитро Зінов'єв

Харківський національний університет імені В.Н. Каразіна, майдан Свободи 4, Харків, 61022, Україна
konstantyn.diachenko@gmail.com, zinoviev@karazin.ua

Рецензент: Ткачук Микола Вячеславович, доктор технічних наук, професор, Харківський національний університет імені В.Н. Каразіна, майдан Свободи, 4, Харків, 61022, Україна.
tka.mobile@gmail.com

Надійшло: Листопад 2019.

Анотація: *Емуляція процесів інтеграції програмних систем із використанням HTTP/HTTPS протоколів на теперішній час широко використовується при розробці програмних систем з розподіленою архітектурою. У роботі розглянуто сучасний стан проблеми, проаналізовано існуючі інструментальні засоби для її вирішення, виявлені недоліки та запропонований концептуальний підхід та вимоги до програми, яка дозволить автоматизувати і налагодити процеси розробки та тестування програмних систем із використанням HTTP/HTTPS протоколів.*

Ключові слова: *комп'ютерні науки; програмні системи; прикладний програмний інтерфейс; REST; емуляція веб-сервісів.*

Вступ

На даний час багато сучасних програмних систем мають розподілену архітектуру, тобто складаються з двох або більше компонентів, які узгоджено взаємодіють між собою.

Для забезпечення якості під час розробки програмного продукту обов'язково виконується його тестування. Тести виконують перевірки відповідності заявлених до програмної системи або її компонентів вимог і реально реалізованої функціональності, здійснювані шляхом спостереження за її або їх роботою в штучно створених ситуаціях і на наборі тестів, створених на основі вимог [1].

В залежності від масштабу та бюджету проекту, розробники відповідних програмних систем (ПС), використовують як мінімум, два середовища для їх розгортання. Це середовище для розробки (DEV) та середовище для готової версії системи (Production). Також, іноді додається третє середовище – для тестування (QA).

Середовище для тестування містить версію продукту, яка планується до запуску на Production середовищі і яка містить зміни відносно до поточної версії, що запущена на Production. На цьому середовищі може проводитися E2E (*end two end*) тестування, тестування прикладного програмного інтерфейсу (ППІ), інтеграційне тестування або мануальне тестування та регресія. Якщо ПС, що розробляється, під час проходження тест-кейсів взаємодіє зі сторонніми системами або веб-сервісами (*компонентами*), які є платні, – плата може стягуватися за кожне звернення, що значно збільшує кошти на розробку. Окрім того, не всі сторонні системи та сервіси надають DEV та Production середовища.

На сьогодні зазначені труднощі вирішують двома способами. Перший, це відключення взаємодії ПС, що розробляється, зі сторонньою службою. Це призводить до зниження якості продукту із-за неможливості тестування процесів взаємодії зі сторонніми службами. Другий спосіб, це емуляція поведінки, сценаріїв або проходження потоків даних сторонньої системи або сервісів [2]. Такий спосіб потребує створення нових системи-емуляторів, розробка яких може займати багато часу.

Тести виконують перевірки, в яких використовуються попередньо накопичені вхідні дані. Наприклад, коли відомо, що на «Запит А» система повинна повернути «Відповідь А». Пара «Запит А» та «Відповідь А» - є вхідними даними для тесту. Для забезпечення достовірності

проходження тест-кейсів необхідно ізолювати усі компоненти, взаємодія яких тестується, бо будь-яка паралельна взаємодія з компонентами ПС може вплинути на вихідні дані. Наприклад, для тестування резервації конкретного продукту треба забезпечити такі умови для тесту, щоб більше ніхто не зміг зарезервувати цей продукт в один і той же час. Цю особливість вирішують відповідні системи-емулятори [3].

Часто буває так, що UI/UX команда змушена розробляти свою частину ПС спираючись на специфікації, контракт або інтерфейс до сервісу, з якими їх компоненти системи будуть взаємодіяти. Із-за відсутності належних інструментів емуляції процесів інтеграції ПС виникають затримки розробки. Також затримки розробки можуть виникати, коли під час розробки та тестування ПС один із сервісів, що необхідний для коректної роботи системи, не працює.

Тому команди розробників і тестувальників змушені витрачати ресурси для створення специфічних інструментів для вирішення проблем, що перераховані вище. Тому, імітація реального ППІ веб-сервісу зараз широко використовується у сфері розробки програмних продуктів. Під імітацією ППІ ми розуміємо керовану емуляцію поведінки ППІ веб-сервісу. Загалом можна зазначити, що імітація реальних сервісів та створення керованих заглушок активно використовується у наступних областях:

- організація E2E, інтеграційного, ППІ, мануального, регресійного, компонентного тестування програмного продукту;
- організація паралельної роботи фронтенд і бекенд команд над однією задачею без затримок;
- ізолювання єдиного сервісу для проведення налагодження програмної системи.

Поширення ідей створення та використання інструментів імітації реальних сервісів обумовлене наступними причинами:

- неможливістю забезпечити ізоляцію середовища від зовнішнього середовища під час перевірки тест-кейсами;
- відсутністю підтримки декількох середовищ стороннім сервісом, з яким взаємодіє система;
- наявністю проблем з проведенням тестування розподілених програмних систем, де система має зв`язки з однієї та більше сторонніми системами або сервісами;
- наявністю проблем з організацією паралельної розробки одного продукту;
- тенденцією на оптимізацію розробки та тестування програмного продукту;
- вимогами до покращення якості кінцевого програмного продукту.

Для більшості розробників та тестувальників, особливий інтерес представляють можливості створення «заглушок» або емуляторів сторонніх систем для використання останніх у процесі інтеграції з програмною системою, що розробляється, та її налагодженням за час, у декілька разів менший у порівнянні зі створенням специфічних інструментів, а також для проведення повного тестування системи із самостійно-керованими системами-емуляторами.

Наявність такого інструменту дозволить:

- заощадити кошти у разі, коли імітована система є платною і не має спеціальних тестових режимів;
- заощадити час на розробку і налагодження певної частини програмного забезпечення;
- ізолювати ПС від стану сторонніх сервісів для тестування бізнес-логіки;
- розгортати інструмент як окремий компонент інфраструктури продукту.

У межах даної роботи розглядається питання стосовно концептуального підходу до розробки інструментальних засобів для емуляції процесів інтеграції програмних систем із використанням HTTP/HTTPS протоколів.

1 Процес емуляції реального прикладного програмного інтерфейсу сервісів або служб

Веб-сервіс (або веб-служба) – це програмний сервіс із стандартизованими інтерфейсами, що ідентифікується веб-адресою. Веб-служби можуть взаємодіяти одна з одною і зі сторонніми додатками за допомогою повідомлень, заснованих на певних протоколах. На сьогодні

для взаємодії компонентів розподіленого додатка в мережі широко використовується архітектурний стиль REST (Representational State Transfer) [4].

Реалізація REST стилю виконується за допомогою протоколу передачі даних – HTTP, який використовується для управління та передачі інформації ресурсу RESTful сервісу.

Імітація реального ППІ дозволяє емулювати будь-який RESTful сервіс.

Це корисно в сценаріях тестування, паралелізації роботи та ізолювання однієї служби.

Для тестування можливо:

- легко відтворювати всі типи відповідей для компонентів складних програмних систем, таких як веб-служби REST;
- ізолювати перевірку системи, щоб переконатися, що тести надійно виконуються і не виконуються, коли існує справжня помилка. Важливим є лише перевірка системи, що тестується, а не її залежностей, щоб уникнути невдач тестів через невідповідні зовнішні зміни, такі як відмова мережі або перезавантаження сервера, де розташована веб-служба;
- легко встановлювати шаблонні відповіді незалежно для кожного тесту, щоб гарантувати наявність тестових даних відповідно до контракту веб-сервісу в кожному тесті;
- створювати тестові твердження, які мають перевіряти запити, які надсилала система, що тестується [5].

При паралелізації роботи можливо наступне:

- почати роботу з ППІ служб до того, як веб-служба буде реалізована. Якщо ППІ або послуга ще не повністю розроблені, емуляція може імітувати ППІ, що дозволяє будь-якій команді, яка використовує цю службу, розпочати роботу без затримки;
- ізолювати групи розробників під час початкових етапів розробки, коли ППІ служби можуть бути надзвичайно нестабільними і мінливими. Використання імітації реальних ППІ дозволяє продовжувати роботу з розробки, навіть коли зовнішня служба виходить з ладу.

Під час розгортання та налагодження ПС буває корисно запускати окрему програму або одну службу, або обробляти підмножину запитів на локальному комп'ютері в режимі налагодження. Використовуючи імітації реального ППІ, легко вибірково пересилати запити до локальних процесів, які виконуються в режимі налагодження.

2 Аналіз інструментів для емуляції прикладного програмного інтерфейсу

На теперішній час існує клас інструментальних засобів, які дозволяють вирішувати задачі імітації процесів інтеграції ПС із використанням HTTP/HTTPS протоколів. До них відносяться, наприклад: MockServer, Mockable.io, Prism, Restmock, та Okhttpmockwebserver.

Розглянемо більш детально можливості та недоліки деяких з них.

MockServer – безкоштовний інструмент для створення заглушок будь-якої системи, яка інтегрується через HTTP або HTTPS [6].

Перш за все цей інструмент дає можливість:

- запускати спеціальний сервер для імітування сервісів і служб для інтеграційного тестування;
- створювати імітацію ППІ реального сервісу і налаштувати проксі до реального сервісу;
- створювати очікувані відповіді за кількома різними характеристиками запиту до сервісу;
- запам'ятовувати відповіді від сервісу;
- підтримки двох мов програмування.

До недоліків MockServer можна віднести наступне:

- необхідність досвіду та знань з програмування для створення емуляторів;
- відсутність графічного інтерфейсу для налаштувань;
- імітація повністю потребує написання програмного коду;
- відсутність можливості збереження накопичених відповідей від реального сервісу;

- відсутність можливості по препроцесингу і постпроцесингу запиту та відповіді.

Програма Mockable.io – безкоштовний інструмент для створення «заглушок» для ППІ. Mockable.io дає можливості створювати очікувані відповіді за явно вказаними характеристиками запиту, динамічно генерувати відповіді від сервісу та має графічний інтерфейс для налаштувань.

До її недоліків можна віднести неможливість зберігання накопичених відповідей від реального сервісу, відсутність можливості щодо препроцесинга і постпроцесинга запиту та відповіді, неможливість створювати очікувані відповіді за кількома різними характеристиками запиту.

Програма Prism – це безкоштовний набір пакетів для створення макетів API з OpenAPI v2 (*раніше відомий як Специфікація Swagger*) та OpenAPI v3.

Переваги Prism:

- можливість створювати очікувані відповіді за явно вказаними характеристиками запиту;
- повне логування;
- створення шаблонів запитів за заданими характеристиками запиту.

Недоліки Prism:

- неможливо зберегти накопичені відповіді від реального сервісу;
- відсутні можливості по препроцесингу і постпроцесингу запиту та відповіді;
- відсутність можливості створювати очікувані відповіді за кількома різними характеристиками запиту;
- відсутність графічного інтерфейсу для налаштувань.

Аналіз можливостей існуючих інструментів для імітації ППІ дозволяє зробити висновок, що існуючі рішення мають або вузький спектр можливостей, або не зручний інтерфейс, або є платними. У реальності все це призводить до того, що частіше за все, задачі, які потребують імітацію зовнішніх сервісів, розробники ПС вирішують за допомогою дрібних самостійно створених програмних рішень. Тому було б корисно створити інструментальний засіб для емуляції процесів інтеграції ПС, який би об'єднав усі корисні функції та був вільний від перелічених недоліків.

3 Концептуальне проектування програми для емуляції прикладного програмного інтерфейсу

По-перше, у функціях програми для імітації прикладного програмного інтерфейсу необхідно передбачити можливість створення декількох режимів емуляції програмної системи. А саме:

- “збір даних” – отримання варіантів відповідей від реального сервісу та їх збереження;
- “проксінг” – звернення безпосередньо до даних реального сервісу;
- “заглушка” – використання збережених або завантажених даних;
- змішаний режим зі сценаріями – препроцесінг, постпроцесінг запитів з використання DataFlow шаблонів на об'єктно-орієнтованій мові Groovy.

Діаграма варіантів використання, що представлена на рис.1, описує систему на концептуальному рівні. Вона відображає відношення між актором “Тестувальник” та прецедентами .

Це дасть можливість розробникам ПЗ створювати гнучкі емуляції веб-служб і використовувати їх як у якості примітивних заглушок так і у якості емуляторів з поведінкою, встановленою Groovy скриптами. Тобто для тестування можна використовувати режим “заглушки”, який дає можливість імітувати веб-сервіси із попередньо завантаженими даними.

Режими “збір даних” та “проксінг” доповнюють один одного, та відрізняються лише особливістю збереження даних і можуть бути використані для підготовки тестування.

Змішаний режим надає можливість задати звичайній заглушці різну поведінку в залежності від зовнішніх параметрів стану системи. Наприклад, коли на один і той же запит ППІ сервіс повинен відповідати по різному в залежності від часу. Для змішаного режиму необхідно мати досвід у написанні скриптів на мові програмування Groovy.

По-друге, необхідно передбачити можливість узгодження запитів за параметрами, тілом або заголовками запиту і запам'ятовувати схожі відповіді для однакових по заданим характеристикам запитів. Іншими словами, для запитів, з однаково визначеними характеристиками, має повертатися однакова відповідь.

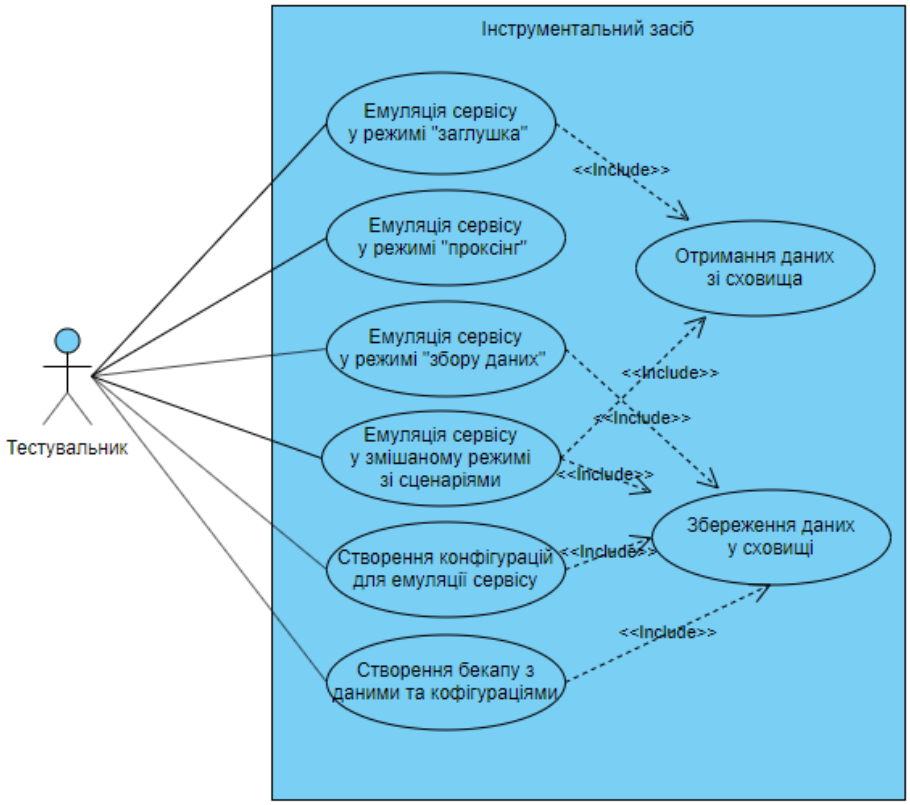


Рис. 1 – Use-case діаграма для інструментального засобу

Втілення такої пропозиції дасть можливість розробнику створювати детальні емуляції веб-служб з урахуванням не тільки URI адреси до ресурсу, а й інших елементів HTTP запиту: заголовків та тіла запиту. Чим більше характеристик може бути залучено під час емуляції веб-сервісу, тим детальніше буде описаний ППІ веб-сервісу для подальшого тестування.

Третя пропозиція, яка має бути реалізована при розробці програми для імітації ППІ – це здатність створювати та задавати сценарії для запитів за допомогою мови XML або за допомогою предметно-орієнтованої мови. Сценарій в контексті даного питання – це опис кроків, які повинні виконуватися під час обробки запиту та відповіді (див. Рис. 2). Можливі варіанти: виклики підзапитів, модифікація будь-яких властивостей запиту, препроцесінг запиту, постпроцесінг запиту, модифікація відповіді.

```
def headerName = 'x-app'
def headerValue = 'mock-service'
request?.headers.find { mockServiceHeader -> mockServiceHeader.name == headerName }.value = [headerValue]
if (request.query) {
    request.query = "${request.query}&lang=en"
}
```

Рис. 2 – Groovy сценарій для модифікації вхідного запиту

Ця пропозиція може бути реалізована за допомогою використання паттерну DataFlow та існуючих реалізацій від компанії Apache. Це надасть можливість розробнику задати поведінку веб-сервісу при його емуляції у випадках, коли неможливо заздалегідь створити дані для

тестування. Наприклад, коли веб-сервісу у запиті передається електронна пошта будь-якого користувача системи та номер картки лояльності. Так як сервіс відповідає на запит, шукаючи за необхідними характеристиками дані для відповіді у заздалегідь підготовлених даних, то для кожного тесту необхідно підготувати дані з новими користувачем. Це дасть можливість створювати моделі для препроцесінгу запиту та постпроцесингу відповіді за допомогою написання Groovy скрипту [7].

Четверта пропозиція полягає у створенні можливості експорту та імпорту даних до інструменту імітації за визначеними шаблонами, а також можливість експорту усієї конфігурації інструменту та можливість імпорту його повної конфігурації без втрат. Це дозволить переносити інструментальний засіб в інші середовища без втрати даних і конфігурацій. У такому разі усі накопичені дані мають зберігатися примусово, або у «бекапі» за розкладом.

4 Висновки

В межах даної роботи були запропоновані концептуальний підхід та відповідні вимоги до інструментального засобу, який дозволить автоматизувати та налагодити процеси розробки та тестування програмних систем із використанням HTTP/HTTPS протоколів.

В цілому, така програма має являти собою систему, яку можна запустити на окремому сервері або локальному комп'ютері (*як окремий компонент з графічним інтерфейсом та REST ППІ*) для емуляції програмних систем.

Загальні вимоги до інструментального засобу можна сформулювати наступним чином:

- інструментальний засіб повинен мати декілька режимів емуляції програмної системи;
- інструментальний засіб повинен надавати можливості орієнтуватися на різні характеристики запиту в залежності від природи запитів до веб-служби;
- інструментальний засіб повинен надавати можливість запам'ятовувати відповіді від сервісу та конфігурації сервісів-емуляторів;
- інструментальний засіб повинен надавати можливість препроцесінгу і постпроцесінгу запитів та відповідей за допомогою сценаріїв, які можуть бути створені на основі предметно-орієнтованої мови Groovy;
- повинен надавати можливість експортувати та імпортувати дані до інструменту емуляції.

Посилання

[1] Кулямин В.В. Место тестирования среди методов оценки качества ПО / В.В. Кулямин, О.Л. Петренко. URL: https://software-testing.ru/library/5-testing/117-2008-10-13-19-25-13 (дата звернення 02.11. 2019)

[2] Aggarwal K. API Mocks and why should we care / Kapil Aggarwal. URL: https://medium.com/@kapil.aggarwal1/api-mocks-and-why-should-we-care-418b24c35c25 (дата звернення 08.11. 2019)

[3] Bulaty W. Stubbing, Mocking and Service Virtualization Differences for Test and Development Teams. URL: https://www.infoq.com/articles/stubbing-mocking-service-virtualization-differences/ (дата звернення 10.11. 2019)

[4] Артемий . Архитектура REST / Артемий. URL: https://habr.com/ru/post/38730/ (дата звернення 06.11. 2019)

[5] Fowler M. Mocks Aren't Stubs / Martin Fovler. URL: https://martinfowler.com/articles/mocksArentStubs.html (дата звернення 12.11. 2019)

[6] Bloom J. D. MockServer. What is Mock Server / James D Bloom. URL: http://www.mock-server.com/#what-is-mockserver (дата звернення 10.11. 2019)

[7] Integrating Groovy in a Java application. URL: http://docs.groovy-lang.org/latest/html/documentation/guide-integrating.html (дата звернення 06.11. 2019)

Reviewer: Tkachuk Mykola, Doctor of Science (Engineering), Full Prof., V.N. Karazin Kharkiv National University, Kharkiv, Ukraine.
E-mail: tka.mobile@gmail.com

Received on November 2019.

Authors:
Kostiantyn Diachenko, student, V. N. Karazin Kharkiv National University, Kharkiv, Ukraine.
E-mail: konstantyn.diachenko@gmail.com.

Dmytro Zinoviev, Senior Lecturer, V. N. Karazin Kharkiv National University, Kharkiv, Ukraine.
E-mail: zinoviev@karazin.ua.

Requirements for creating tools to emulate software integration processes using HTTP/HTTPS protocols.

Abstract. Emulation of software systems integration processes using HTTP/HTTPS protocols is now widely used in the development of distributed systems software systems. The paper discusses the current state of the problem, analyzes the existing tools to solve it, identifies shortcomings, and offers a conceptual approach and program requirements that will automate and debug software development and testing processes using HTTP/HTTPS protocols.

Keywords: Computer Science; Software Systems; Application Programming Interface; REST; Emulation of Web Services.

Рецензент: Ткачук Николай Вячеславович, д.т.н., проф., Харьковский национальный университет имени В. Н. Каразина, Харьков, Украина.
E-mail: tka.mobile@gmail.com.

Поступила: Ноябрь 2019.

Авторы:
Константин Дьяченко, студент 6 курса, Харьковский национальный университет имени В. Н. Каразина, Харьков, Украина.
E-mail: konstantyn.diachenko@gmail.com

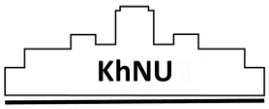
Дмитрий Зиновьев, ст. преподаватель, Харьковский национальный университет имени В. Н. Каразина, Харьков, Украина.
E-mail: zinoviev@karazin.ua

Требования к созданию инструментальных средств для эмуляции процессов интеграции программных систем с использованием HTTP/HTTPS протоколов.

Аннотация. Эмуляция процессов интеграции программных систем с использованием HTTP/HTTPS протоколов в настоящее время широко используется при разработке программных систем с распределенной архитектурой. В работе рассмотрено современное состояние проблемы, проанализированы существующие инструментальные средства для ее решения, выявленные недостатки и предложен концептуальный подход и требования к программе, которая позволит автоматизировать и наладить процессы разработки и тестирования программных систем с использованием HTTP/HTTPS протоколов.

Ключевые слова: компьютерные науки; программные системы; прикладной программный интерфейс; REST; эмуляция веб-сервисов.

ISSN 2519-2310 DOI: 10.26565/2519-2310-2019-2



Наукове видання

КОМП'ЮТЕРНІ НАУКИ ТА КІБЕРБЕЗПЕКА

Випуск 2(14) 2019

Міжнародний електронний науково-теоретичний журнал

Англійською, українською, російською мовами

Комп'ютерне верстання – Федоренко В.В., Єсіна М.В.

61022, Харків, майдан Свободи, 6
Харківський національний університет імені В.Н. Каразіна

V. N. Karazin Kharkiv National University Publishing

2019