УДК 004.42

РОЗРОБКА ПЛАГІНА ДЛЯ ВІЗУАЛІЗАЦІЇ СТРУКТУРНИХ СХЕМ ОБЧИСЛЮВАЧІВ НА ОСНОВІ ТЕКСТОВОГО ОПИСУ АЛГОРИТМІВ ГАРМОНІЧНИХ ПЕРЕТВОРЕНЬ

Процько І. – д-р техн. наук, професор, кафедра автоматизованих систем управління, Національний університет «Львівська політехніка», Львів, Україна.

Теслюк В. – д-р техн. наук, професор, кафедра автоматизованих систем управління, Національний університет «Львівська політехніка», Львів, Україна.

АНОТАЦІЯ

Актуальність. У багатьох областях науки і техніки чисельне рішення задач недостатньо для подальшого розвитку реалізацій отриманих результатів. Серед існуючих підходів візуалізації інформації вибирають той, який дозволяє ефективно здійснити розкриття неструктурованих дієвих ідей, узагальнити або спростити аналіз отриманих даних. Результати візуалізації узагальнених структурних схем на основі текстового опису алгоритму наочно відображають взаємодію його частин, що важливо на системотехнічному етапі проєктування обчислювачів.

Мета дослідження – наліз та програмна реалізація візуалізації структури на прикладі обчислювачів дискретних гармонічних перетворень, отриманих в результаті синтезу алгоритму на основі циклічних згорток з можливістю розширення візуалізації структур на інші обчислювальні алгоритми.

Метод. Узагальнена схема синтезу алгоритмів швидких гармонічних перетворень у вигляді набору операцій циклічної згортки над комбінованими послідовностями вхідних даних і коефіцієнтами гармонічної функції перетворення з візуалізацією їх у вигляді узагальненої структурної схеми обчислювача.

Результати. Результатом роботи є програмна реалізація візуалізації узагальнених структурних схем для синтезованих алгоритмів косинусного та Хартлі перетворень, що наочно відображають взаємодію основних блоків обчислювача. Програмна реалізація візуалізації структури обчислювача виконана на мові TypeScript з використанням фреймворку Phaser 3.

Висновки. У роботі розглянуто та проаналізовано розроблену програмну реалізацію візуалізації загальної структури обчислювача для швидких алгоритмів дискретних гармонічних перетворень в області дійсних чисел, отриманих в результаті синтезу алгоритму на основі циклічних згорток. Результати візуалізації варіантів структурних схем обчислювачів наочно і зрозуміло відображають взаємодію його частини і дозволяють виконати оцінку того чи іншого варіанту обчислювального алгоритму в процесі проєктування.

КЛЮЧОВІ СЛОВА: комп'ютерна візуалізація, рендеринг, структурна схема, плагін візуалізації, гармонічні перетворення.

АБРЕВІАТУРИ

ISO/IEC – International Organization of Standardization/International Electrotechnical Commission;

ITU-T – International Telecommunication Union – Telecommunication sector;

UML – Unified Modeling Language;

ДКП – дискретне косинусне перетворення;

ДПХ – дискретне перетворення Хартлі;

ТМ – твірний масив;

ЦЗ – циклічні згортки.

НОМЕНКЛАТУРА

$a(n)$ – коефіцієнт зміщення;

h_{ij} – цілочисельний елемент твірного під масиву;

$H(L)$ – твірний масив;

$H_i(L_i)$ – твірний підмасив;

$Hr(n_1)$ – твірний масив рядків;

$Hc(n_2)$ – твірний масив стовпців;

k – кількість підмасивів у твірному масиві;

L – обсяг твірного масиву;

L_i – обсяг підмасиву;

N – обсяг перетворення;

T – період базисної функції;

$x(m)$ – вхідна послідовність перетворення;

$X^c(n)$ – вихідна послідовність косинусного перетворення;

$X^h(n)$ – вихідна послідовність перетворення Хартлі.

ВСТУП

Комп'ютерна візуалізація використовує конкретні процеси графічної побудови, які визначаються сферою застосувань [1]. У багатьох областях науки і техніки чисельне рішення задач недостатньо для подальшого розвитку реалізацій отриманих результатів. Для цього використовують різноманітні підходи візуалізації інформації, які дозволяють ефективно здійснити розкриття неструктурованих дієвих ідей, узагальнити або спростити аналіз отриманих даних. Сучасні можливості візуального представлення інформації дозволяють інтерпретувати дані, використовуючи процедури побудови графіків поверхонь, створення масивів даних для тривимірної графіки, вмикання і вимикання масштабної сітки, керування властивостями осей графіків та інші [2]. Ці засоби відображення результатів алгебраїчної аналітики можуть полегшити розуміння складних

DOI 10.15588/1607-3274-2024-2-15

концепцій і прийняття рішень для конкретних дослідників і розробників.

Об'єктом дослідження є процес візуалізації загальної структури обчислювача дискретних гармонічних перетворень в області дійсних чисел, отриманих в результаті синтезу алгоритму на основі ЦЗ. Описано процес отримання швидкого алгоритму у вигляді тексту, в який включає набір операцій ЦЗ над послідовностями вхідних даних та коефіцієнтами базисної функції перетворення.

Предметом дослідження є спосіб візуалізації загальних структурних схем обчислювачів ДКП та ДПХ на основі ЦЗ через рендеринг структури обчислювача з використанням конвольверів.

Метою дослідження є аналіз та програмна реалізація візуалізації узагальненої структури на прикладі обчислювачів дискретних гармонічних перетворень, отриманих в результаті синтезу алгоритмів на основі ЦЗ з можливістю розширення візуалізації структур на інші обчислювальні алгоритми.

Результати візуалізації варіантів структурних схем обчислювача для конкретного обсягу гармонічного перетворення наочно і зрозуміло відображають взаємодію його частини і дозволяють виконати їх аналіз на системотехнічному етапі проєктування.

1 ПОСТАНОВКА ПРОБЛЕМИ

За текстовим описом відповідної структури (Рис. 3) в результаті синтезу алгоритму ДКП та ДПХ на основі ЦЗ обсягу N з $x(m)$, m=0, 1,…N–1, що містить елементи SuperWrap з кількістю від 1 до N/2. SuperWrap складається з одного або більше Wrap елемента a), b),..., які представляють ЦЗ позначену (X). Кожен Wrap складається з складових: Sum Set, Summation Set. Складова Sum Set містить набір елементів Signed Number, що є знаковими цілочисельними аргументами базисної функції перетворення в межах від 0 до (N–1). Складова Summation Set складається з набору типу Summation Operand Set, який характеризується знаком («+» або «–»), а також набором $x(m)$, m=0, 1,…N–1, елементів Summation Operand.

Виконати візуалізацію загальної структурної схеми обчислювача за текстовим описом, що міститиме p-точкові конвольвери з відповідними входами, а виходи яких об'єднуються для визначення $X^c(n)$ або $X^h(n)$, n=0,1,…N–1. Користувацький інтерфейс реалізувати з вимогою мінімалізму та досягненням економії процесорного часу на промальовку, до позначень на структурній схемі включити текстове пояснення.

2 ОГЛЯД ЛІТЕРАТУРИ

Існує багато пакетів візуалізації та мов програмування і програмних середовищ для відображення у графічному вигляді розроблених обчислювальних алгоритмів [3]. Деякі з них зовсім прості: потрібно тільки завантажити дані та вибрати спосіб відображення. Інші програми більш складні і комплексні — вимагають налаштувань і відповідних знань для відображення особливостей отриманих рішень.

Наукові розробки з ефективних обчислень та сучасні системи генерації швидких алгоритмів описують їх у вигляді послідовності алгебраїчних операцій над вхідними даними до отримання кінцевих результуючих даних. Подавати інформацію так, щоб вона виділяла певні особливості взаємозв'язків серед інших в аналітичному методі є достатньо складним завданням. Для цього використовують у тому ж програмному середовищі, де реалізовано обчислювальний алгоритм, відповідні програмні засоби інфографіки з метою чіткого структурованого відображення комплексної інформації [4]. Тобто, результати синтезу ефективних обчислювальних алгоритмів потребують своєї інтерпретації не тільки у вигляді блок-схем, але і в узагальненому структурованому вигляді. Це потрібно для детальнішого аналізу та на системотехнічному етапі проєктування різноманітних інформаційних систем.

В багатьох роботах підкреслюється, що для розуміння особливостей отриманих результатів, необхідно виділити та вибрати необхідні об'єкти за допомогою візуалізації, яка наочно і зрозуміло відображає елементи та варіанти у вигляді структурних схем. Поряд з тим досліджуються питання простоти, гнучкості та масштабованості візуалізації великих структур даних [5]. В роботах [6, 7] відзначається, що краще розуміння концепції структури даних і алгоритмів полягає в посиленні основ об'єктно-орієнтованого програмування. У роботі [8] реалізовано візуалізацію різноманітних алгоритмів за допомогою модулів PyGame та Tkinter python, де крок за кроком відображається алгоритм і як різним алгоритмам потрібен різний час для виконання завдань. В багатьох роботах акцентується, що використання різноманітних інформаційних технологій забезпечує краще розуміння роботи обчислювального алгоритму та визначення його складності.

3 МАТЕРІАЛИ ТА МЕТОДИ

Для представлення інформаційних даних в спектральний образ поряд з перетворенням Фур'є використовуються алгоритми дійсного дискретного косинусного або синусного перетворення, дискретного перетворення Хартлі. Дані перетворення знаходять своє застосування в інформаційних технологіях різноманітного призначення в тому числі у згорткових нейронних мережах. Обчислення дискретних перетворень класу Фур'є являється однією з найбільш тривалих процедур в інформаційних технологіях [9]. Розроблено ряд підходів, що дозволять зменшити обчислювальну складність і, відповідно, пришвидшити роботу

DOI 10.15588/1607-3274-2024-2-15

програмного та апаратного забезпечення дискретних гармонічних перетворень. Існують ефективні алгоритми обчислення для одно, дво та багатовимірних дискретних перетворень класу Фур'є, які називають швидкими перетвореннями. Багатоваріантність ефективних обчислень розділяють на алгоритми з основою два, розщепленою основою, змішаною основою, непарного обсягу, складеного обсягу і алгоритм простих множників. Для візуального відображення цих швидких перетворень, що в більшості відповідають алгоритмам типу Кулі-Тюкі, широко використовують потоки графів з базовою операцією у вигляді так званого «метелика».

Одним з ефективних підходів є використання ЦЗ для обчислення дискретних перетворень класу Фур'є [10]. На відміну від структурних схем у вигляді потокових графів, алгоритми обчислення дискретних перетворень класу Фур'є на основі ЦЗ для свого візуального відображення потребують інші базові операції. Серед цих основних базових операцій є дії поелементного об'єднання послідовностей даних, обчислення *p*-точкових ЦЗ, результати виконання яких відповідним чином поелементно з'єднуються між собою. Для обсягів перетворень класу Фур'є, що розкладається на велику кількість простих множників, в швидких алгоритмах міститься велика кількість цих операцій. Це вимагає оптимізації їх розміщення з бажаними семантичними або візуальними кореляціями в остаточному макеті візуалізації структури обчислювача.

Організаціями ISO/IEC та ITU-T стандартизовано вісім видів дискретного косинусного перетворення, що знаходять широке застосування в процесі обробки інформаційних сигналів [11]. Широко застосовують ДПХ (1) для виконання перетворення в області дійсних значень даних в їх спектральний образ

$$X^{h}(n) = \frac{1}{N}\sum_{m=0}^{N-1} cas\left[\frac{2\pi\, n\, m}{N}\right]x(m) = \\ = \sum_{m=0}^{N-1} c(n,m)x(m), \quad n = 0,1,\ldots, N-1, \tag{1}$$

де $c(n,m)$=cas $(2\pi nm/N)$ = cos$(2\pi nm/N)$+sin$(2\pi nm/N)$.

На відміну від перетворення Фур'є, що відображає дійсні функції у комплексну область, перетворення Хартлі відображають дійсні вхідні дані у дійсний образ, використовуючи базисну функцію, що представляє собою суму косинуса і синуса одного аргументу [12].

Ці тригонометричні перетворення є подальшим розвитком дискретних перетворень Фур'є, що виконуються в дійсній області. Одним з підходів ефективного обчислення дискретних перетворень є приведення їх гармонічного базису до вигляду блочно-циклічних матричних структур з подальшим обчисленням перетворень за допомогою швидких ЦЗ [13]. В роботі [14] описується приведення гармонічного базису перетворення ДКП до набору циклічних зліва підматриць з використанням твірних масивів.

В результаті застосування підходу, структуру базисної матриці можна задати твірним масивом

$$H(L) = H_1(L_1)H_2(L_2)\ldots H_k(L_k) = \\ = (h_{11}, h_{12}, \ldots h_{1L1})(h_{21}, h_{22}, \ldots h_{2L2})\ldots(h_{kL1}, h_{kL2}, \ldots h_{kLk}). \tag{2}$$

Обсяг твірного масиву L дорівнює сумі обсягів підмасивів Li

$$L = (L_1 + L_2 + \ldots + L_k). \tag{3}$$

Цілочисельні елементи h_{ij} твірного підмасиву $H_i(L_i)$, які є аргументами базисної гармонічної функції перетворення (i=1,2,…,k; j=1,2,…,L_i) та $h_{ij} < T$ менші періоду повторення базисної гармонічної функції. Розглянемо синтез швидкого алгоритму для ДКП-II, яке просто називають дискретним косинусним перетворенням і вивели на основі дискретного перетворення Фур'є. Пряме ДКП-II представлено у вигляді:

$$X^{c}(n) = a(n)\sum_{m=0}^{N-1}\cos\left[\frac{n(2m+1)\pi}{2N}\right]x(m) = \\ = \sum_{m=0}^{N-1} c(n,m)x(m), \quad n = 0,1,\ldots, N-1, \tag{4}$$

де $c(n, m) = \cos(n(2m+1)\pi/2N)$.

Всі рядки $c(n,m)$ векторів $[c(k,0),\ldots,c(k,N-1)]$ є ортогональні і нормалізовані за виключенням першого. Коефіцієнт $a(n)$ зводить їх до ортонормальності

$$a(n) = \begin{cases}\sqrt{1/N}, & n = 0\\ \sqrt{2/N}, & n = 1,2,\ldots,N-1.\end{cases} \tag{5}$$

Твірний масив $H(L)$ визначається за підстановкою з рядків/стовпців аргументів гармонічної функції базисної матриці перетворення обсягу N. На основі твірного масиву здійснюється переіндексація рядків/стовпців базисної матриці, що в результаті приводить до формування блочно-циклічних матричних структур в базисній матриці ДКП. Внаслідок різних виразів індексів стовпців та рядків, що входять до аргументів базисної функції (4) для переіндексації використовуються твірний масив $Hr(n_1)$ рядків та твірний масив $Hc(n_2)$ стовпців.

Отже, одержуємо матрицю розмірності ($n_1 \times n_2$), що містить набір цілочисельних циклічних зліва підматриць різних обсягів, де розмірності $n_1 = 2N$, $n_2 = N$ визначаються обсягом твірних масивів для рядків та стовпців. Кожна з циклічних квадратних підматриць містить цілочисельні елементи, які

DOI 10.15588/1607-3274-2024-2-15

належать одному з твірних підмасивів $H_i(L_i)$. Обсяги L_i кожного з твірних підмасивів залежать від множників розкладу обсягу перетворення N, і в сумі відповідають умові (3).

Для дослідження структури базисних матриць ДКП розроблено універсальний програмний засіб аналізу цілочисельних матриць, який виконує сканування всього набору елементів блочно-циклічної матриці. Для пошуку заданого фрагменту за максимальною шириною і висотою L_i, в матриці виконується покрокове сканування з переміщенням згори вниз і зліва направо.

Для синтезу швидкого алгоритму ДКП необхідно виконати аналіз структури одержаної блочно-циклічної матриці з метою визначення ідентичних блоків, що розміщених горизонтально та вертикально один відносно іншого. Наявність ідентичних блоків приводить до зменшення обчислювальної складності та надає можливість розпаралелення виконання обчислення ДКП. В процесі автоматичного синтезу алгоритмів обчислення ДКП довільного обсягу N на основі ЦЗ на завершальному етапі аналізу блочно-циклічних матричних структур базисної матриці ДКП виконується визначення інформаційних даних про кількість ідентичних циклічних підмасивів та їх розташування в базисній матриці.

Ідентичні циклічні підматриці, що розміщені вертикально один відносно іншого приводять до одноразового обчислення циклічних згорток, результати яких в процесі об'єднання використовуються для визначення вихідних значень перетворення $X^c(m)$.

Ідентичні циклічні підматриці, що розміщені горизонтально один відносно іншого приводять до об'єднання груп вхідних значень $x(n)$ перетворення і одноразового обчислення ЦЗ. Результати ЦЗ в процесі об'єднання використовуються для однієї групи вихідних значень перетворення $X^c(m)$. Виконання поелементних додавань вхідних значень $x(n)$ будуть використовуватись для однотипових циклічних підматриць, розміщених по горизонталі.

Вихідні значення $X^c(m)$ перетворення ДКП отримують об'єднанням результатів згорток по горизонталі на основі відповідних координат.

Приклад виконання синтезу швидкого алгоритму ДКП для обсягу N=15 з твірним масивом $H_r(n_1)$ рядків та твірним масивом $H_c(n_2)$ стовпців мають вигляд

$H_r(30)$ = (0) (1 7 11 17) (29 23 19 13) (2 14 22 26) (28 16 8 4) (3 21 27 9) (5 25) (6 18) (24 12) (10) (20);
$H_c(15)$ = (0) (1 7 11 13) (14 8 4 2) (3 9) (12 6) (5) (10).

Результат (рис. 1) формування блочно-циклічних матричних структур аргументів базисної матриці ДКП для обсягу N=15 з твірними масивами $H_r(30)$ та $H_c(15)$.

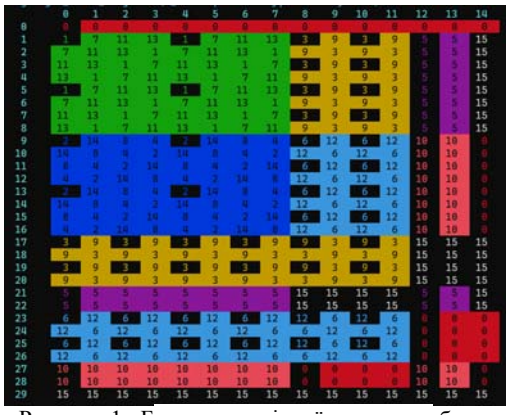

Рисунок 1– Блочно-циклічної структури базисної матриці ДКП для обсягу N=15

В результаті аналізу блочно-циклічної структури базисної матриці формується текстовий файл про кількість ідентичних циклічних підмасивів в базисній матриці (рис. 1). Ці дані дозволяють визначити кількість та обсяг ЦЗ, можливість їх паралельного обчислення та об'єднання обчислених значень ЦЗ для визначення вихідних значень ДКП. Текстовий файл ДКП-II для обсягу N=15 має вигляд

1. a) (+0) (X) { +x(0) +x(3) +x(5) +x(8) +x(14) +x(11) +x(9) +x(6) +x(1) +x(10) +x(13) +x(4) +x(2) +x(12) +x(7) }

2. a) (+1 +7 +11 –13) (X) { +(x(0), x(3), x(5), x(8)) –(x(14), x(11), x(9), x(6)) }
b) (+3 –9) (X) { +(x(1), x(10)) –(x(13), x(4)) }
c) (+5) (X) { +x(2) –x(12) }

3. a) (+2 +14 –8 –4) (X) { +(x(0), x(3), x(5), x(8)) +(x(14), x(11), x(9), x(6)) }
b) (+6 –12) (X) { +(x(1), x(10)) +(x(13), x(4)) }
c) (+10) (X) { +x(2) +x(12) }
d) (–0) (X) { –x(7) }

4. a) (+3 –9) (X) { +(x(0), x(3)) –(x(5), x(8)) –(x(14), x(11)) +(x(9), x(6)) }
b) (+9 +3) (X) { +(x(1), x(10)) –(x(13), x(4)) }

5. a) (+5) (X) { +x(0) –x(3) +x(5) –x(8) –x(14) +x(11) –x(9) +x(6) –x(2) +x(12) }

6. a) (+6 –12) (X) { +(x(0), x(3)) +(x(5), x(8)) +(x(14), x(11)) +(x(9), x(6)) }
b) (–12 +6) (X) { –(x(1), x(10)) –(x(13), x(4)) }
c) (–0) (X) { –x(2) –x(12) –x(7) }

7. a) (+10) (X) { +x(0) +x(3) +x(5) +x(8) +x(14) +x(11) +x(9) +x(6) +x(2) +x(12) }
b) (–0) (X) { –x(1) –x(10) –x(13) –x(4) –x(7) }

Обсяг та кількість ЦЗ для ДКП з N=15 дорівнює: 1– точкова згортка: 8; 2– точкова згортка: 6; 4– точкова згортка: 2.

В роботі [15] описується приведення гармонічного базису перетворення ДПХ до набору циклічних зліва

DOI 10.15588/1607-3274-2024-2-15

підматриць з використанням твірних масивів. Розглянемо приклад виконання синтезу швидкого алгоритму ДКП для обсягу N=15 з твірним масивом $H(N)$, що має вигляд

$$H(15) = (0)\ (1\ \ 2\ \ 4\ \ 8)\ (14\ \ 13\ \ 11\ \ 7)\ (3\ \ 6\ \ 12\ \ 9)\ (5\ \ 10).$$

Результат (рис. 2) формування блочно-циклічних матричних структур аргументів в базисній матриці ДПХ для обсягу N=15 з твірними масивами $H(15)$

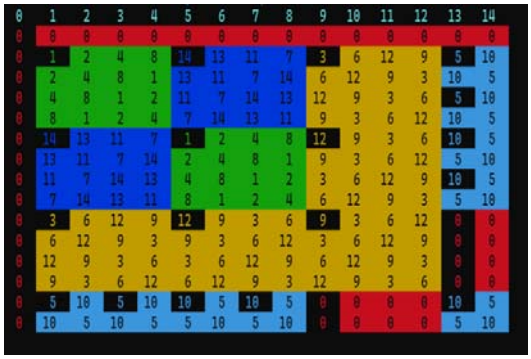

Рисунок 2 – Блочно-циклічної структури базисної матриці ДПХ для обсягу N=15

В результаті аналізу блочно-циклічної структури базисної матриці формується текстовий файл про кількість ідентичних циклічних підмасивів в базисній матриці (рис 2). Текстовий файл ДПХ для обсягу N=15 має вигляд

1. a) (+0) (X) { +x(0) +x(1) +x(2) +x(4) +x(8) +x(14) +x(13) +x(11) +x(7) +x(3) +x(6) +x(12) +x(9) +x(5) +x(10) }

2. a) (+0) (X) { +x(0) }
   b) (+1 +2 +4 –8) (X) { +(x(1), x(2), x(4), x(8)) }
   c) (+14 –13 –11 –7) (X) { +(x(14), x(13), x(11), x(7)) }
   d) (+3 –6 –12 –9) (X){+(x(3), x(6), x(12), x(9)) }
   e) (+5 –10) (X) { +(x(5), x(10)) }

3. a) (+0) (X) { +x(0) }
   b) (+14 –13 –11 –7) (X) { +(x(1), x(2), x(4), x(8)) }
   c) (+1 +2 +4 –8) (X) { +(x(14), x(13), x(11), x(7)) }
   d) (–12 –9 +3 –6) (X) {–(x(3), x(6), x(12), x(9)) }
   e) (–10 +5) (X) { –(x(5), x(10)) }

4. a) (+0) (X) { +x(0) +x(5) +x(10) }
   b) (+3 –6 –12 –9) (X) { +(x(1), x(2), x(4), x(8)) }
   c) (–12 –9 +3 –6) (X) { –(x(14), x(13), x(11), x(7)) }
   d) (–9 +3 –6 –12) (X) {–(x(3), x(6), x(12), x(9)) }

5. a) (+0) (X) { +x(0) +x(3) +x(6) +x(12) +x(9) }
   b) (+5 –10) (X) { +(x(1), x(2)) +(x(4), x(8)) }
   c) (–10 +5) (X) { –(x(14), x(13)) –(x(11), x(7)) –(x(5), x(10)) }

Обсяг та кількість ЦЗ для ДПХ з N=15 дорівнює: 1– точкова згортка: 4; 2– точкова згортка: 4; 4– точкова згортка: 9.

4 ЕКСПЕРИМЕНТИ

Для програмної реалізації візуалізації структури обчислювача вибрано мову програмування TypeScript. Мова TypeScript розширює свою попередницю JavaScript, забезпечуючи зворотну сумісність. Мова TypeScript є статично типізованою, що дасть можливість виявити помилки ще на етапі написання коду або компіляції. Для роботи з TypeScript (яка є розробкою Microsoft) найбільше підходять середовища розробки компанії Microsoft: Visual Studio та Visual Studio Code. Середовище Visual Studio є більш «важким» рішенням, адже працює повільніше, проте й має більше функціоналу. Середовище ж Visual Studio Code створено спеціально для роботи з веб-додатками і є надзвичайно простим, розширюваним з великою кількістю плагінів.

Для роботи з версіями програми використовується популярна і сильно розровсюджена система Git. Через свою розповсюдженість та простоту для керування пакетами використовуватиметься *npm*. Для пакування модулів використано Webpack, як стандарт в сучасній індустрії веб-додатків. Використання *npm* та Webpack передбачає використання Node.js. Для роботи з файлами (зчитування та запис) використовуватиметься бібліотека «file-saver». Для розроблення модульних тестів (unit-тестів) використовуватиметься набір бібліотек: chai, mocha, sinon.

Вибір інструментів для візуалізації. Найбільш популярними рішеннями для забезпечення візуалізації є: Pixi.js, Phaser 2, Phaser 3. Перший інструмент є дуже швидким, проте має лише базові функції, тому розроблення буде дещо повільнішим, ніж в інших варіантах. Оскільки швидкість виконання не є найважливішим атрибутом якості системи візуалізації, що розробляється, то розглянемо інші варіанти. Phaser 2 – це популярне рішення, що має велику кількість функцій, і з яким легко працювати. Проте даний фреймворк є дещо застарілим і не підтримується розробниками. Phaser 3 – фреймворк, що прийшов на заміну свого попередника Phaser 2, і є повним переосмисленням його. Дане рішення стало новим стандартом в індустрії і активно розвивається. Тому інструментом для візуалізації вибрано Phaser 3.

Початковими даними для роботи системи візуалізації є текстовий файл алгоритму у результаті аналізу блочно-циклічної структури базисної матриці. Текстовий файл формується за відповідною структурою представлення даних швидкого алгоритму обчислення. На рис. 3 зображено схему, яка на прикладі демонструє те, на які складові структуруються дані текстового опису алгоритму.

DOI 10.15588/1607-3274-2024-2-15

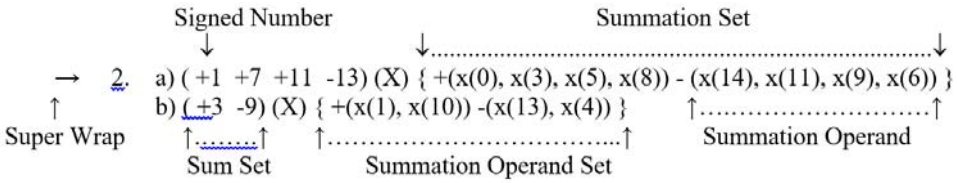

Рисунок 3 – Шаблон структур даних

Позначення (X) означає виконання операції, у даному випадку ЦЗ над послідовностями базисних функцій з аргументами поданими на початку у круглих дужках ( ) та послідовностями вхідних даних перетворення x(*i*), *i*=0,1,...*N*–1, які необхідно попередньо поелементно об'єднати.

Головним блоком (рис. 3) є елемент Super Wrap, який відповідає за одну згортку. Він складається з одного або більше елемента a), b), ... Wrap, які представляють ЦЗ. Кожен Super Wrap складається з складових: Sum Set, Summation Set. Складова Sum Set містить набір елементів Signed Number. Складова Summation Set складається з набору типу Summation Operand Set, який характеризується знаком («+» або «–»), а також набором елементів Summation Operand.

Програма візуалізації структури обчислювача використовує модулі: токенізер, парсер, оптимізатор, візуалізатор.

Токенізатор опрацьовує «сирі» символьні дані з текстового файлу опису і розбиває їх на послідовність токенів. В алгоритмі загальної логіки модуля-токенізатора виконуються повторювані дії доки не буде досягнуто кінця даних. Серед дій, що виконуються є зчитування символу і, в залежності від його типу, виконується відповідна підпроцедура для зчитування токена (токени можуть складатись з кількох символів). Крім того, виконується перевірка, чи підтримується зчитаний символ. Отриманий набір токенів передається на синтаксичний аналіз доки не буде досягнуто кінця даних.

Перед синтаксичним аналізом окремої згортки поділяється вхідний набір токенів на групи, які формують певну ієрархію. Потреба в цьому алгоритмі спричинена тим, що формат вхідних даних передбачає ієрархічність, де на першому рівні є чисельні індекси («1.», «2.» тощо), а на другому – символьні («а)», «б)» тощо). Даний алгоритм розділяє токени спершу на чисельно-індексовані групи, а потім кожну групу поділяє на символьно-індексовані підгрупи. Потім ці групи та підгрупи використовуютьсяв модулі синтаксичного аналізатора.

Синтаксичний аналізатор отримує токени і перетворює їх на синтаксичне дерево. Цей процес деколи називають парсингом. Тобто з текстового файлу опису згорток в алгоритмі синтаксичного аналізу зчитується кожен рядок і використовуються відповідні граматичні правила для аналізу лише однієї згортки. Далі виконується зчитування першої послідовності згортки, роздільника (X), другої послідовності. Далі перевіряється, чи залишились дані. Якщо так, то формується синтаксичне дерево, інакше – отримуємо помилку.

Наступним модулем є оптимізатор, завданням якого є, дослідивши структуру отриманого дерева, зробити певні перетворення, які б сформували дерево більш компактним та зручним для візуалізації. Оптимізація на даний момент є реалізована лише базово і буде допрацьовуватись в наступних версіях програмної системи.

Модуль візуалізатора здійснює інфографіку структурної схеми ДКП. Для цього виконується проста лінійна послідовність кроків:

– виконується підрахунок параметрів, що допоможуть виконати візуалізацію (він здійснюється на основі синтаксичного дерева);

– відображаються прямокутники та підписи, що їх стосуються;

– відображаються різноманітні стрілки та їх підписи.

Таким чином, оптимізоване синтаксичне дерево, а також налаштування, отримані з користувацького інтерфейсу, надходять у візуалізатор. Він створює візуальне зображення структури обчислювача.

На рис. 4 зображено UML-діаграму класів системи для візуального відображення структурної схеми обчислювача. UML-діаграму класів формує уявлення про те, якою є статична структура програмного забезпечення системи.

Варто також зазначити, що на даній схемі відображено лише основні ключові класи, методи та поля. Насправді система міститиме більше дрібних об'єктів, серед яких, зокрема, перелічення, інтерфейси, деякі не надто важливі допоміжні класи для синтаксичного аналізу, конфігурації, утилітарні класи тощо.

DOI 10.15588/1607-3274-2024-2-15

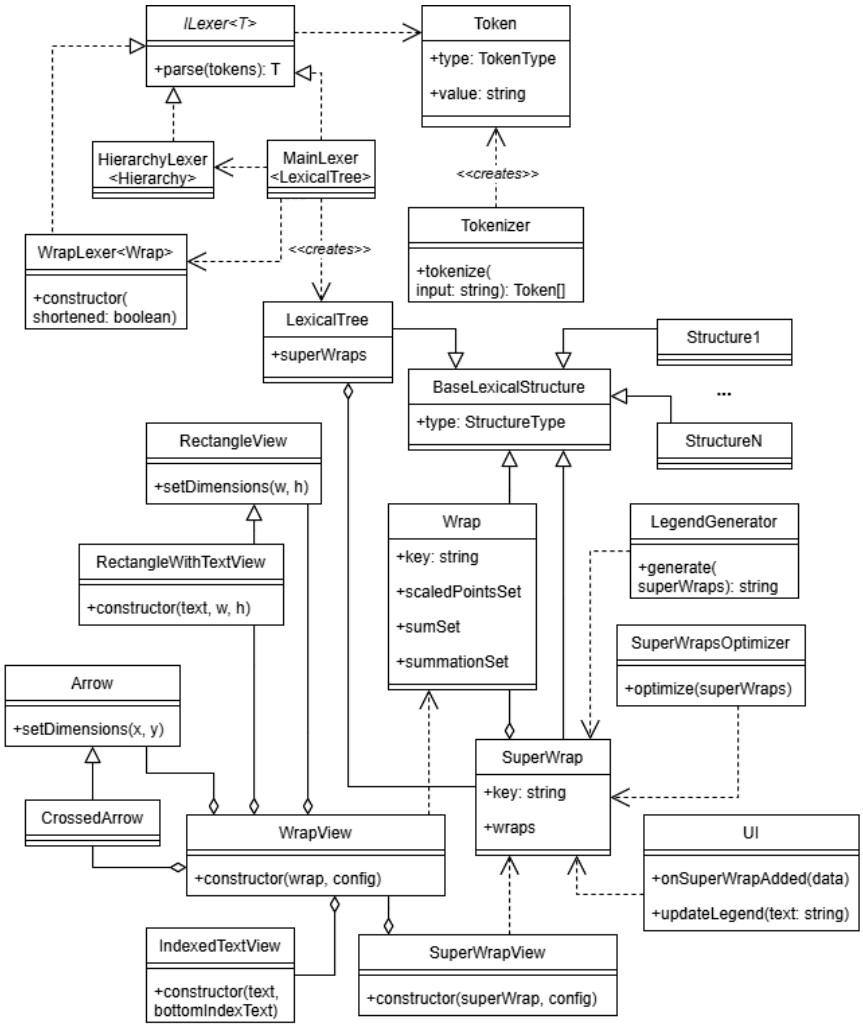

Рисунок 4 – Діаграма класів системи візуалізації

5 РЕЗУЛЬТАТИ

Взаємодія з системою здійснюється через користувацький інтерфейс, у якому проводяться налаштування для відображення структурної схеми (Рис. 5). В результаті отримуємо відображення системи на екрані, а також можливість завантажити зображення в файли, що міститимуть структурну схему обчислювача або частину.

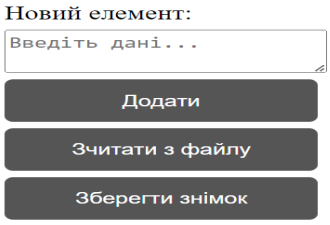

Рисунок 5 – Користувацький інтерфейс для візуалізації структури обчислювача

На рис. 6 бачимо приклад того, як виглядає відображення структури обчислювача ДКП на основі текстового файлу ДКП-II для обсягу N=15. Структура складається з набору стрілок (вхідних, проміжних, вихідних), підписів до них (як от x'_{2a}), та прямокутників (CC та U). CC є блоками p-точкових ЦЗ, U є блоками поелементного об'єднання вхідних даних $x(i)$, $X(i)$ вихідні дані перетворення, i=0,1,...,N–1.

Дане зображення передається на користувацький інтерфейс, який в свою чергу, відображає все на екран для користувача, а також дає можливість зберегти отримане зображення у файл. Крім цього, можна побачити текстове пояснення до позначень на схемі

$x'_{2a}=x'_{3a}=(x(0), x(3), x(5), x(8))$
$x''_{2a}=x''_{3a}=(x(14), x(11), x(9), x(6))$
$x'_{2b}=x'_{3b}=x'_{4b}=x'_{6b}=(x(1), x(10))$
$x''_{2b}=x''_{3b}=x''_{4b}=x''_{6b}=(x(13), x(4))$
$x'_{2c}=x(2)$
$x''_{2c}=x(12)$
$x'_{3c}=(x(2), x(12))$
$x'_{3d}=x(7)$
$x'_{4a}=x'_{6a}=(x(0), x(3), x(14), x(11))$
$x''_{4a}=x''_{6a}=(x(5), x(8), x(9), x(6))$
$x'_{5a}=(x(0), x(5), x(14), x(9), x(2))$
$x''_{5a}=(x(3), x(8), x(11), x(6), x(12))$
$x'_{6c}=(x(2), x(12), x(7))$
$x'_{7a}=(x(0), x(3), x(5), x(8), x(14), x(11), x(9), x(6), x(2), x(12))$
$x'_{7b}=(x(1), x(10), x(13), x(4), x(7))$

DOI 10.15588/1607-3274-2024-2-15

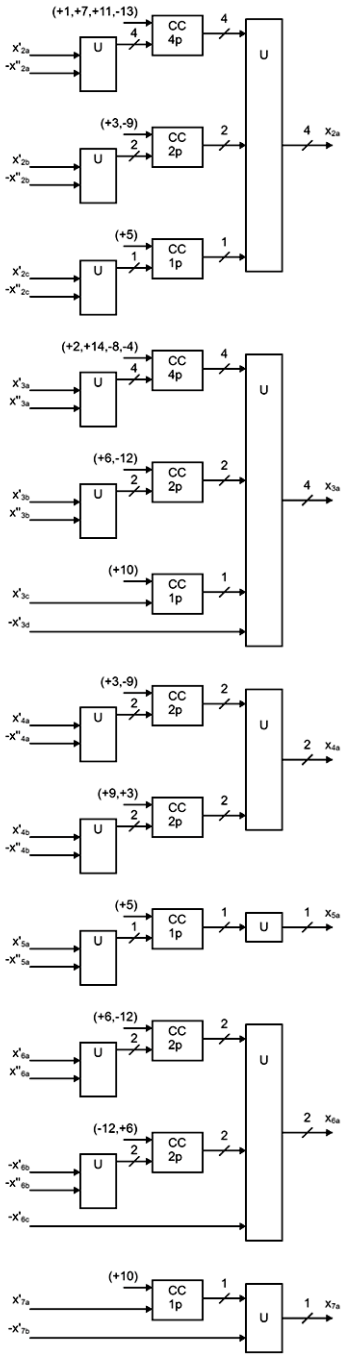

Рисунок 6 – Візуальне зображення структури обчислювача ДКП обсягу N=15

На рис. 7 бачимо результат того, як виглядає частина відображення структури обчислювача ДПХ обсягу N=15 на основі тексту

5. a) (+0) (X) { +x(0) +x(3) +x(6) +x(12) +x(9) }
b) (+5 –10) (X) { +(x(1), x(2)) +(x(4), x(8)) }
c) (–10 +5) (X) { –(x(14), x(13)) –(x(11), x(7)) –(x(5), x(10)) }

з текстового файлу ДПХ для обсягу N=15.

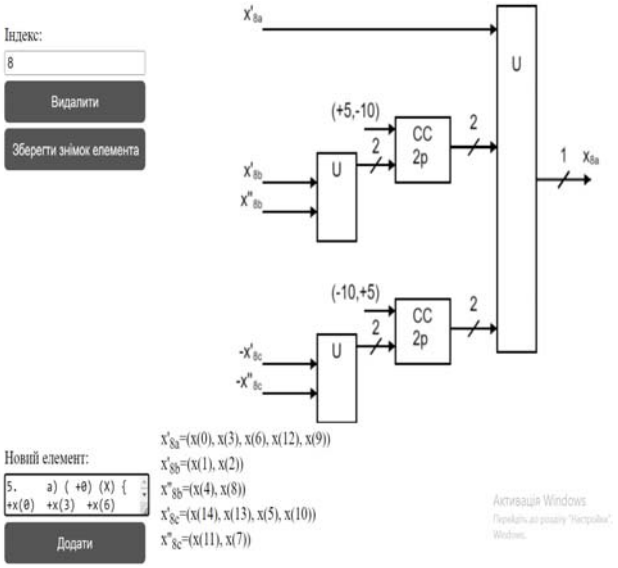

Рисунок 7 – Візуальне зображення частини структури обчислювача ДПХ обсягу N=15

Модуль генерації текстового пояснення (рис. 7) так само, як і візуалізатор, використовує оптимізоване синтаксичне дерево, та передає результати своєї роботи на вхід до користувацького інтерфейсу.

6 ОБГОВОРЕННЯ

Програма синтезу алгоритмів швидких гармонічних перетворень на основі ЦЗ завдяки отриманому текстовому опису алгоритму включає розроблений плагін для візуалізації структурних схем обчислювачів. Це дає можливість швидко оцінити кількість конвольверів їх взаємозв'язок, обсяги виконання ЦЗ, послідовності об'єднання вхідних даних на системотехнічному етапі проєктування обчислювача. Реалізація обчислювачів швидких гармонічних перетворень на основі ЦЗ у вигляді інтегральних схем має ряд переваг, що характеризуються високим показником параметрів площа затримка [16].

В плагіні застосовано технології Web з метою досягнення простоти розроблення та підтримки, кросплатформності. Використано провідну технологію Phaser 3 для реалізації візуалізації векторних примітивів. Для того, щоб система не мала таку шкідливу властивість, притаманну деяким програмним рішенням, як «сильну зв'язність», вона чітко розділена на дві частини. Ядро системи, частина *core*,, відповідає за логіку програмного застосунку і виконує:

– токенізацію;
– синтаксичний аналіз;
– оптимізацію.

Частина *view*, що має відношення до відображення будь-чого на екрані, включає:

– графічний користувацький інтерфейс;
– візуалізаційний модуль для згорток.

Оскільки система є насиченою алгоритмами, то легко можна допустити помилки, які буде важко

DOI 10.15588/1607-3274-2024-2-15

виправити, коли розробка досягне великого масштабу. Для того щоб «відловити» та виправити помилки, було прийнято важливе рішення – під час розроблення системи розробляти також й модульні тести (unit tests). Загалом написано 12 тестів, які перевіряють коректність 5 класів, які стосуються найважливішої частини логіки системи – токенізація та синтаксичний аналіз.

В ході розроблення, неодноразово було помічено, що результати виконання тестів свідчать про наявні помилки у виконанні програми, і, таким чином, було легко вирішено всі проблеми, які виникали через випадково допущені помилки в коді.

Система має мінімально необхідний набір функцій, тому може розширюватись, зокрема шляхом покращення оптимізації лексичного дерева. Також, розроблену програмну візуалізацію можна покращити для більш інтерактивного та більш гнучкого налаштування на візуалізацію структур різноманітних обчислювачів.

ВИСНОВКИ

У роботі розглянуто візуалізацію загальної структури обчислювача, отриманого в результаті синтезу алгоритмів швидких гармонічних перетворень на основі ЦЗ.

Наукова новизна роботи полягає у визначенні особливостей структур обчислювачів на основі загального відображення через конвольвери, які візуалізують опис синтезованих блочно-циклічних базисних матриць дискретних гармонічних перетворень. На основі створеного опису алгоритму, що включає багатоярусний набір ЦЗ, розроблена програма візуалізації структури обчислювача дискретних гармонічних перетворень. Програма візуалізації реалізована на мові програмування TypeScript з використанням фреймворка Phaser 3 для візуалізації векторних примітивів. Результати візуалізації структурних схем обчислювача ДКП і ДПХ для конкретного обсягу перетворення наочно і зрозуміло відображають взаємодію його частини, що важливо на системотехнічному етапі проєктування обчислювача.

Для розроблення застосовано технології Web, досягнувши таким чином простоту розроблення, підтримки, кросплатформність. Користувацький інтерфейс реалізовано з вимогою мінімалізму з досягненням економії процесорного час на промальовку. Дизайн є контрастним та з приглушеними спокійними кольорами: білий, чорний, відтінки сірого. Кросплатформність надає можливість працювати з системою без встановлення додаткового програмного забезпечення.

Практичне значення роботи полягає у тому, що розроблений плагін для візуалізації загальних структурних схем обчислювачів ДКП і ДПХ може використовуватись для відображення загальних структур і інших обчислювальних алгоритмів, який дозволить наочно і зрозуміло відобразити взаємодію його частини та виконати їх аналіз на системотехнічному етапі проєктування.

Напрям подальших досліджень полягатиме в розробці алгоритмічного та програмного забезпечення що забезпечить аналіз та вибір обчислювальних структур за відповідними критеріями на основі різноманітних варіантів синтезованих швидких алгоритмів обчислення дискретних гармонічних перетворень певного типу та обсягу.

ПОДЯКИ

Робота виконана в рамках держбюджетної науково-дослідної роботи «Методи та засоби інтелектуального вимірювання параметрів руху та визначення просторової орієнтації наземних мобільних робототехнічних платформ» національного університету «Львівська політехніка».

ЛІТЕРАТУРА

1. Дрєєв О. М. Аналіз комп'ютерних систем візуалізації з метою алгоритмізації обґрунтування щодо їх використання / О. М. Дрєєв, Б. Ю. Железняк // Конструювання, виробництво та експлуатація сільськогосподарських машин. – 2021. – Вип. 51. – С. 227–238. DOI: 10.32515/2414-3820.2021.51.227-238
2. Xue M. Research on Information Visualization Graphic Design Teaching Based on DBN Algorithm / M. Xue // Computational Intelligence and Neuroscience. – 2021. – Vol. 6. – P. 1–10. DOI: 10.1155/2021/3355030
3. 36 кращих інструментів для візуалізації даних. – Access mode: https://toplead.com.ua/ua/blog/id/38-luchshih-instrumentov-dlja-vizualizacii-dannyh-160/
4. Performance evaluation and analysis of sparse matrix and graph kernels on heterogeneous processors / [F. Zhang, W. Liu, N. Feng et al.] // CCF Transactions on High Performance Computing. – 2019. – Vol. 1. – P. 131–143
5. Garriga J. Towards a comprehensive visualization of structure in data / J. Garriga, F. Bartumeus // arXiv. – 2021. – Access mode: https://arxiv.org/abs/2111.15506?context=cs
6. Simonak S. Increasing the Engagement Level in Algorithms and Data Structures Course by Driving Algorithm Visualizations / S. Simonak // Informatica. – September 2020. – Vol. 44, Issue 3, DOI: 10.31449/inf.v44i3.2864
7. Ghadge S. A Survey paper on data structure and algorithm visualization / S. Ghadge, V. Mane // International Research Journal of Modernization in Engineering Technology and Science. – April-2022. – Vol. 04, Issue 04. – P. 232–236
8. Gupta A. S. Algorithm Visualization / A. S. Gupta, M. Vyawahare, A. Viz // 5th Biennial International Conference on Nascent Technologies in Engineering (ICNTE). – Navi Mumbai, India, 20–21 January 2023. – P. 1–5. DOI: 10.1109/ICNTE56631.2023.10146719
9. Oppenheimer A. V. Discrete-Time Signal Processing. Third edition. Englewood Cliffs / A. V. Oppenheimer, R. W. Schafer. – NJ : Prentice Hall. Pearson Education Limited, 2014. –1042 p.
10. Blahut R. E. Fast algorithms for signal processing / R. E. Blahut. – Cambridge : University Press, 2010. – 469 p. DOI: 10.1017/CBO9780511760921
11. MPEG-4, MPEG-4: ISO/IEC 14496-2:2004. Information technology – Coding of audio-visual objects – Part 2: Visual, ISO, 2004.
12. Dziech A. New Orthogonal Transforms for Signal and Image Processing / A. Dziech // Applied Sciences. – 2021. – Vol. 11, Issue 16. – P. 7433. DOI: 10.3390/app11167433

DOI 10.15588/1607-3274-2024-2-15

13. Prots'ko I. Block-Cyclic Structuring of the Basis of Fourier Transforms Based on Cyclic Substitution / I. Prots'ko, M. Mishchuk // Cybernetics and Systems Analysis. – 2021. – Vol. 57, Issue 6. – P. 1008–1016.
14. Prots'ko I. Algorithm of Efficient Computation of DCT I–IV Using Cyclic Convolutions / I. Prots'ko // International Journal of Circuits, Systems and Signal Processing. – 2013. – Vol. 7, Issue 1. – P. 1–9.
15. Prots'ko I. Algorithm of efficient computation of generalized discrete Hartley transform based on cyclic convolutions / I. Prots'ko // IET Signal Processing. – 2014. – Vol. 8, Issue 4. – P. 301–308.
16. Chiper D. F. An Efficient Algorithm and Architecture for the VLSI Implementation of Integer DCT That Allows an Efficient Incorporation of the Hardware Security with a Low Overhead / D. F. Chiper, A. Cracan // Applied Sciences. – 2023, Vol. 13, Issue 12, 6927. DOI: 10.3390/app13126927

Стаття надійшла до редакції 09.02.2024.
Після доробки 25.04.2024.

UDC 004.42

DEVELOPMENT OF A PLUG-IN FOR VIZUALIZATION OG STRUCTURAL SCHEMES OF COMPUTERS BASED ON THE TEXTUAL DESCRIPTION OF ALGORITHMS OF HARMONIC TRANSFORMS

Prots'ko I. – Dr. Sc., Professor, Department of Automated Control Systems, Lviv National Polytechnic University, Lviv, Ukraine.
Teslyuk V. – Dr. Sc., Professor, Department of Automated Control Systems, Lviv Polytechnic National University, Lviv, Ukraine.

ABSTRACT

Context. In many areas of science and technology, the numerical solution of problems is not enough for the further development of the implementation of the obtained results. Among the existing information visualization approaches, the one that allows you to effectively reveal unstructured actionable ideas, generalize or simplify the analysis of the received data is chosen. The results of visualization of generalized structural diagrams based on the textual description of the algorithm clearly reflect the interaction of its parts, which is important at the system engineering stage of computer design.

Objective of the study is the analysis and software implementation of structure visualization using the example of discrete harmonic transformation calculators obtained as a result of the synthesis of an algorithm based on cyclic convolutions with the possibility of extending the structure visualization to other computational algorithms.

Method. The generalized scheme of the synthesis of algorithms of fast harmonic transformations in the form of a set of cyclic convolution operations on the combined sequences of input data and the coefficients of the harmonic transformation function with their visualization in the form of a generalized structural diagram of the calculator.

The results. The result of the work is a software implementation of the visualization of generalized structural diagrams for the synthesized algorithms of cosine and Hartley transformations, which visually reflect the interaction of the main blocks of the computer. The software implementation of computer structure visualization is made in TypeScript using the Phaser 3 framework.

Conclusions. The work considers and analyzes the developed software implementation of visualization of the general structure of the calculator for fast algorithms of discrete harmonic transformations in the domain of real numbers, obtained as a result of the synthesis of the algorithm based on cyclic convolutions. The results of visualization of variants of structural schemes of computers clearly and clearly reflect the interaction of its parts and allow to evaluate one or another variant of the computing algorithm in the design process.

KEYWORDS: computer visualization, rendering, structural scheme, visualization plugin, harmonic transforms.

REFERENCES

1. Drieiev O., Zhelesnya B. Analysis of Computer Visualization Systems in Order to Algorithmize the Rationale for their Use, *Design, production and Exploitation of Agricultural Machines*, 2021, Vol. 51, pp. 227–238. DOI: 10.32515/2414-3820.2021.51.227-238
2. Xue M. Research on Information Visualization Graphic Design Teaching Based on DBN Algorithm, *Computational Intelligence and Neuroscience,* 2021, Vol. 6, pp. 1–10. DOI: 10.1155/2021/3355030
3. 36 best data visualization tools. – Access mode: https://toplead.com.ua/ua/blog/id/38-luchshih-instrumentov-dlja-vizualizacii-dannyh-160/
4. Zhang F., Liu W., Feng N., Zhai J., Du X. Performance evaluation and analysis of sparse matrix and graph kernels on heterogeneous processors, *CCF Transactions on High Performance Computing,* 2019, Vol. 1, pp. 131–143
5. Garriga J., Bartumeus F. Towards a comprehensive visualization of structure in data, *arXiv*, 2021, Access mode: https://arxiv.org/abs/2111.15506?context=cs
6. Simonak S. Increasing the Engagement Level in Algorithms and Data Structures Course by Driving Algorithm Visualizations, *Informatica,* September 2020, Vol. 44, Issue 3, DOI: 10.31449/inf.v44i3.2864
7. Ghadge S., Mane V. A Survey paper on data structure and algorithm visualization, *International Research Journal of Modernization in Engineering Technology and Science*, April-2022, Vol. 04, Issue 04, pp. 232–236
8. Gupta A. S., Vyawahare M., Viz A. Algorithm Visualization, *5th Biennial International Conference on Nascent Technologies in Engineering (ICNTE)*. Navi Mumbai, India, 20–21 January 2023, pp. 1–5. DOI: 10.1109/ICNTE56631.2023.10146719
9. Oppenheimer A. V. Schafer R. W. Discrete-Time Signal Processing. Third edition. Englewood Cliffs. NJ, Prentice Hall, Pearson Education Limited, 2014, 1042 p.
10. Blahut R. E. Fast algorithms for signal processing. Cambridge, University Press, 2010, 469 p. DOI: 10.1017/CBO9780511760921
11. MPEG-4, MPEG-4: ISO/IEC 14496-2:2004. Information technology, Coding of audio-visual objects, Part 2, Visual, ISO, 2004.
12. Dziech A. New Orthogonal Transforms for Signal and Image Processing, *Applied Sciences*, 2021, Vol. 11, Issue 16, P. 7433. DOI: 10.3390/app11167433
13. Prots'ko I., Mishchuk M. Block-Cyclic Structuring of the Basis of Fourier Transforms Based on Cyclic Substitution, *Cybernetics and Systems Analysis*, 2021, Vol. 57, Issue 6, pp. 1008–1016.
14. Prots'ko I. Algorithm of Efficient Computation of DCT I–IV Using Cyclic Convolutions, *International Journal of Circuits, Systems and Signal Processing*, 2013, Vol. 7, Issue 1, pp. 1–9.
15. Prots'ko I. Algorithm of efficient computation of generalized discrete Hartley transform based on cyclic convolutions, *IET Signal Processing*, 2014, Vol. 8, Issue 4, pp. 301–308.
16. Chiper D. F., Cracan A. An Efficient Algorithm and Architecture for the VLSI Implementation of Integer DCT That Allows an Efficient Incorporation of the Hardware Security with a Low Overhead, *Applied Sciences,* 2023, Vol. 13, Issue 12, P. 6927. DOI: 10.3390/app13126927

DOI 10.15588/1607-3274-2024-2-15