Хмельницький національний університет
Факультет інформаційних технологій
Кафедра комп'ютерної інженерії та інформаційних систем

КВАЛІФІКАЦІЙНА РОБОТА

Метод виявлення помилок високої кратності на основі надлишкової системи залишкових класів
Назва теми

Рівень вищої освіти *другий (магістерський)*

Галузь знань *12 «Інформаційні технології»*
Шифр, назва

Спеціальність *123 «Комп'ютерна інженерія»*
Шифр, назва

Освітня програма *«Комп'ютерна інженерія та програмування»*
Назва

Шифр КвРКІ 240116.24.01.15 ПЗ

Виконав здобувач II курсу, група КІ2м-24-1 _____ Олександр КАРПОВ
Підпис Ініціали, прізвище

Керівник д. техн. наук, професор _____ Василь ЯЦКІВ
Науковий ступінь, учене звання Підпис Ініціали, прізвище

Нормоконтролер д. техн. наук, професор _____ Сергій ЛИСЕНКО
Науковий ступінь, учене звання Підпис Ініціали, прізвище

До захисту допускаю:
завідувач кафедри КІІС _____ Ольга ПАВЛОВА
«01» травня 2026 р. Підпис Ініціали, прізвище
дата

Хмельницький 2026

ХМЕЛЬНИЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

Факультет ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

Кафедра КОМП'ЮТЕРНОЇ ІНЖЕНЕРІЇ ТА ІНФОРМАЦІЙНИХ СИСТЕМ

Рівень вищої освіти ДРУГИЙ (МАГІСТЕРСЬКИЙ)

Галузь знань 12 ІНФОРМАЦІЙНІ ТЕХНОЛОГІЇ

Спеціальність 123 КОМП'ЮТЕРНА ІНЖЕНЕРІЯ

Освітня програма «КОМП'ЮТЕРНА ІНЖЕНЕРІЯ ТА ПРОГРАМУВАННЯ»

ЗАТВЕРДЖУЮ
Завідувачка кафедри КІІС
_____ Ольга ПАВЛОВА

" 12 " 01 2026 р.

ЗАВДАННЯ НА КВАЛІФІКАЦІЙНУ РОБОТУ

Карпову Олександру Олеговичу
Прізвище, ім'я, по батькові студента

1. Тема проекту (роботи) Метод виявлення помилок високої кратності на основі надлишкової системи залишкових класів
Керівник проекту (роботи) Яцків Василь Васильович, д.т.н., проф.
Прізвище, ім'я, по батькові, науковий ступінь, вчене звання

Затверджена наказом ректора університету від 12.01.2026 р. № 6

2. Термін подання здобувачем роботи на кафедру 01.05.2026 р.

3. Вихідні дані до роботи Завдання на кваліфікаційну роботу

4. Зміст пояснювальної записки (перелік питань, які потрібно розробити)
Аналіз методів виявлення та корекції помилок у системах залишкових класів
Моделювання процесу виявлення помилок високої кратності на основі надлишкової системи залишкових класів
Метод виявлення помилок високої кратності на основі надлишкової системи залишкових класів
Програмна реалізація та експериментальне дослідження методу виявлення помилок високої кратності на основі надлишкової системи залишкових класів

5. Перелік графічного матеріалу (із зазначенням обов'язкових креслень)

6. Консультанти розділів кваліфікаційної роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв

7. Дата видачі завдання « 12 » 01 2026 р.

КАЛЕНДАРНИЙ ПЛАН

№з/п	Назва етапів (розділів) кваліфікаційної роботи магістра	Термін виконання етапів проєкту (роботи)	Примітка
1	Вибір напряму дослідження та узгодження тематики кваліфікаційної роботи з керівником	12.01.2026	виконано
2	Ознайомлення з предметною областю; формулювання мети та задач дослідження; визначення об'єкта та предмета дослідження	12.01.2026	виконано
3	Робота над розділом 1 – аналіз відомих моделей, методів за темою, постановка задачі	20.01.2026	виконано
4	Робота над розділом 2 – розробка моделей для вирішення поставленої задачі	01.02.2026	виконано
5	Робота над науковими тезами	01.03.2026	виконано
6	Робота над розділом 3 – розробка методів для вирішення поставленої задачі	29.03.2026	виконано
7	Робота над розділом 4 – проектування та розробка методу для вирішення поставленої задачі, експериментальна частина	01.04.2026	виконано
8	Оформлення пояснювальної записки згідно вимог	25.04.2026	виконано
9	Попередній захист ВКР	27.04.2026	виконано
10	Захист ВКР на засіданні ЕК	До 20.05.2026	

Здобувач _____ Олександр КАРПОВ
Підпис Ім'я, ПРІЗВИЩЕ

Керівник кваліфікаційної роботи _____ Василь ЯЦКІВ
Підпис Ім'я, ПРІЗВИЩЕ

РЕФЕРАТ

Тема кваліфікаційної роботи магістра: Метод виявлення помилок високої кратності на основі надлишкової системи залишкових класів.

Автор роботи: Карпов Олександр Олегович

Керівник роботи: Яцків Василь Васильович

Пояснювальна записка: 75 с., 13 рис., 1 табл., 3 дод., 81 джерел.

СИСТЕМА ЗАЛИШКОВИХ КЛАСІВ, НАДЛИШКОВА СИСТЕМА ЗАЛИШКОВИХ КЛАСІВ, ОПТИМІЗАЦІЯ ОБЧИСЛЕНЬ, ВІДНОВЛЕННЯ ДАНИХ, ПОМИЛКИ, ПАКЕТНІ ПОМИЛКИ.

Об’єктом дослідження є процес виявлення та виправлення помилок високої кратності в комп’ютерних системах передачі та обробки даних.

Предметом дослідження є метод виявлення помилок високої кратності на основі надлишкової системи залишкових класів.

Метою кваліфікаційної роботи магістра є підвищення швидкодії та достовірності виявлення помилок високої кратності в комп’ютерних системах на основі надлишкової системи залишкових класів.

Для розв’язання поставлених задач використовувалися методи теорії чисел та модулярної арифметики, системного аналізу, а також методи математичного та імітаційного комп’ютерного моделювання для проведення експериментальних досліджень.

Наукова новизна отриманих результатів:

− набув подальшого розвитку метод виявлення помилок високої кратності на основі надлишкової системи залишкових класів, який, на відміну від відомих класичних методів сліпого перебору проекцій, використовує математичну модель формування вектора підозрілості, кластеризацію суміжних збоїв та дворівневу верифікацію зі швидким математичним фільтром, що дозволяє уникнути комбінаторного вибуху обчислень та локалізувати пакетні помилки за поліноміальний час;

− набули подальшого розвитку програмно-технічні засоби виявлення помилок високої кратності, що дозволяють здійснювати ефективне декодування та успішне відновлення інформації навіть при кратності помилок, що перевищує класичну теоретичну межу виправлення.

Практична значимість отриманих результатів полягає у розробленні архітектури та створенні програмної моделі гібридного декодера (на мові Python), який забезпечує багаторазове прискорення процесу локалізації пакетних помилок порівняно з класичними методами та доводить принципову можливість корекції збоїв високої кратності у високонавантажених обчислювальних системах.

У першому розділі проведено аналіз сучасного стану проблеми забезпечення надійності даних, виявлено алгоритмічні недоліки існуючих методів декодування в НСЗК та обґрунтовано необхідність створення нового методу для боротьби з пакетними помилками високої кратності.

У другому розділі розроблено математичні моделі функціональних блоків системи, описано динамічні діапазони та формалізовано проблему комбінаторного вибуху обчислень і явища аліасингу.

У третьому розділі розроблено гібридний метод локалізації пакетних помилок, що включає формування вектора підозрілості, просторову кластеризацію та дворівневу точну верифікацію із застосуванням швидкого математичного фільтра.

У четвертому розділі здійснено програмну реалізацію гібридного методу та проведено серію експериментальних досліджень, які емпірично довели досягнення мети дослідження та високу швидкодію розробленого рішення.

ЗМІСТ

СКОРОЧЕННЯ ТА УМОВНІ ПОЗНАКИ

СЗК – система залишкових класів

НСЗК – надлишкова система залишкових класів

КТПР – китайська теорема про залишки

ПЛІС – програмована логічна інтегральна схема

IoT – internet of things (інтернет речей)

СЗЗО - системи зі змішаною основою

ВСТУП

У сучасному світі стрімкий розвиток мереж 6G, масове впровадження Інтернету речей (IoT) та хмарних технологій висувають безпрецедентні вимоги до надійності та безперебійності обміну даними. В умовах щільного компонування нанометрових кристалів та впливу інтенсивних електромагнітних завад, наприклад у промислових системах чи мережах VANET, помилки передачі даних дедалі частіше набувають масового, пакетного характеру. Звичайні методи резервування або класичні коригувальні коди часто не справляються з такими навантаженнями через надмірну надлишковість та високу обчислювальну складність процедур відновлення.

Альтернативним та перспективним напрямком є використання непозиційної надлишкової системи залишкових класів. Її здатність до паралельної обробки даних та ізоляції помилок на рівні окремих модулів створює потужний фундамент для побудови відмовостійких систем. Проте існуючі класичні методи декодування в НСЗК стикаються з проблемою комбінаторного вибуху обчислень при виникненні помилок високої кратності, що призводить до неприйнятних затримок і робить їх неефективними для систем реального часу.

Актуальність роботи полягає у необхідності розробки вдосконаленого методу виявлення та виправлення помилок високої кратності в надлишкових системах залишкових класів, який дозволить уникнути комбінаторного зростання обчислювальної складності та забезпечить високу швидкодію локалізації пакетних збоїв.

Метою кваліфікаційної роботи магістра є оптимізація та підвищення швидкодії виявлення помилок високої кратності в комп’ютерних системах на основі надлишкової системи залишкових класів.

Поставлена мета досягається розв’язанням таких основних завдань:

− проаналізувати відомі методи виявлення та корекції помилок у системах залишкових класів та виявити їхні обмеження;

− розробити математичну модель процесу виявлення помилок високої кратності на основі надлишкової системи залишкових класів;

− удосконалити метод виявлення помилок високої кратності на основі надлишкової системи залишкових класів для усунення ефекту алгоритмічного колапсу;

− здійснити програмну реалізацію та експериментальне дослідження розробленого методу виявлення помилок високої кратності.

Об’єктом дослідження є процес виявлення та виправлення помилок високої кратності в комп’ютерних системах передачі та обробки даних.

Предметом дослідження є метод виявлення помилок високої кратності на основі надлишкової системи залишкових класів.

Наукова новизна отриманих результатів:

− набув подальшого розвитку метод виявлення помилок високої кратності на основі надлишкової системи залишкових класів, який, на відміну від відомих класичних методів сліпого перебору проекцій, використовує математичну модель формування вектора підозрілості, кластеризацію суміжних збоїв та дворівневу верифікацію зі швидким математичним фільтром, що дозволяє уникнути комбінаторного вибуху обчислень та локалізувати пакетні помилки за поліноміальний час;

− набули подальшого розвитку програмно-технічні засоби виявлення помилок високої кратності, що дозволяють здійснювати ефективне декодування та успішне відновлення інформації навіть при кратності помилок, що перевищує класичну теоретичну межу виправлення.

На основі проведених досліджень розроблено метод виявлення помилок високої кратності на основі надлишкової системи залишкових класів .

Практична значимість отриманих результатів полягає у розробленні архітектури та створенні програмної моделі гібридного декодера, який забезпечує багаторазове прискорення процесу локалізації пакетних помилок порівняно з класичними методами та доводить принципову можливість корекції збоїв високої кратності у високонавантажених обчислювальних системах.

Для розв’язання поставлених задач використовувалися методи теорії чисел та модулярної арифметики, системного аналізу, а також методи математичного та імітаційного комп’ютерного моделювання для проведення експериментальних досліджень.

За темою кваліфікаційної роботи опубліковано одну публікацію [81] тези у 17-й Міжнародній студентській науковотехнічній конференції «Перспективні мережеві та комп’ютерні технології» – ПЕРСИК 2026 (м. Харків, 23 квіт. 2026). Харків, 2026.

1 АНАЛІЗ МЕТОДІВ ВИЯВЛЕННЯ ТА КОРЕКЦІЇ ПОМИЛОК У СИСТЕМАХ ЗАЛИШКОВИХ КЛАСІВ

1.1 Сучасний стан проблеми забезпечення надійності та завадостійкості даних

Останніми роками спостерігаються кардинальні зміни в глобальних інформаційних системах, які змусили переглянути актуальні підходи до роботи з даними. Перехід до мереж 6G, масове використання Інтернету речей та активний розвиток хмарних технологій сформували нові вимоги до надійності інформаційного обміну. В таких умовах забезпечення та збереження цілісності даних більше не можуть забезпечуватися суто апаратними функціями – потребується додавання до вже наявних апаратних функцій ще й різних математичних функцій, що призводить до ускладнення проєктування та потребує нових рішень.

Наукові дослідження свідчать, що майбутні стандарти зв’язку повинні гарантувати надвисокий рівень точності інформації, яку передають – до 99,9%. Окрім того, також повинні зрости і швидкості обміну цієї інформації [1]. Водночас відомі методи канального кодування, зокрема полярні коди, які продемонстрували ефективність у системах 4G і 5G, поступово досягають меж своїх можливостей.

Проблема надійності на фізичному рівні сучасних обчислювальних платформ є серйозною. Подальше зменшення розмірів напівпровідникових технологій призвело до зростання чутливості мікросхем до зовнішніх факторів, зокрема іонізуючого випромінювання.

Вплив космічних частинок і альфа-випромінювання є однією з ключових причин виникнення м’яких помилок у статичній оперативній пам’яті та програмованих логічних інтегральних схемах [2]. Особливу небезпеку становить те, що в сучасних високощільних мікросхемах одна заряджена частинка може спричинити порушення не окремого біта, а одразу кількох суміжних комірок пам’яті. Такі явища, відомі як багатобітові збої, значно знижують ефективність класичних кодів корекції помилок. Виявляється, що традиційні алгоритми типу

одиночного та подвійного виправляння помилок просто не справляються з множинними помилками, які стають все частішими.

Разом із розвитком мережевих технологій розвивається і підхід до збереження та відновлення даних в системах хмарного збереження даних та середовищах хмарного обчислювання. В хмарних технологіях зазвичай виділяють 3 категорії помилок. Вони перелічені на рисунку 1.1. Звичайне дублювання даних вже не працює як раніше, і на це є кілька вагомих причин – збільшення обсягів інформації, яка в наш час збільшується в геометричній прогресії, що несе за собою економічні втрати для сервісів хмарних середовищ, адже збільшується вартість зберігання інформації. У сучасних дослідженнях пропонують альтернативу класичному – це використання кодів із виправленням стирань, які дозволяють відновлювати втрачені частини даних за рахунок меншої надлишковості [3],[4].

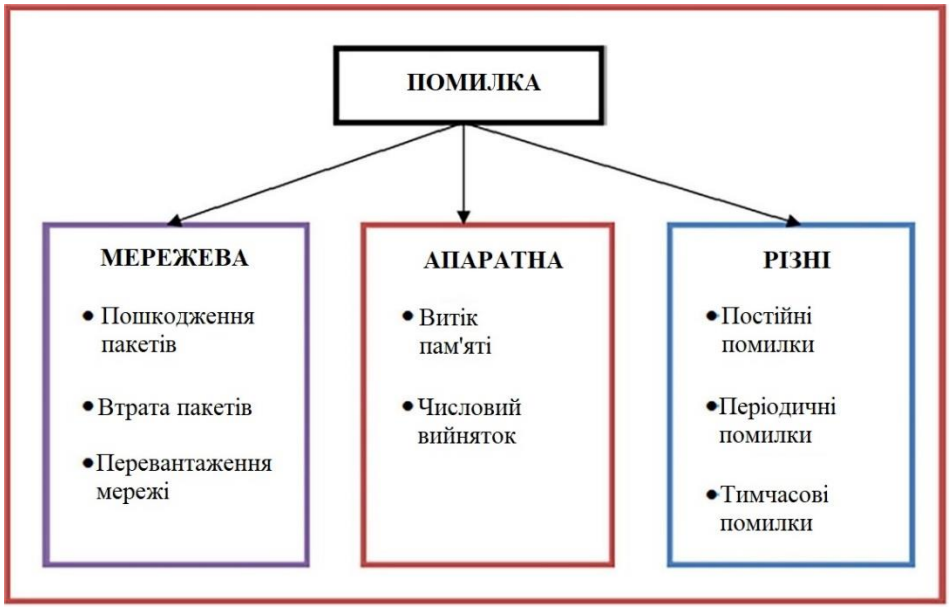


Рисунок 1.1 – Типові помилки в хмарних середовищах та їх категорії [3]

Водночас впровадження надлишкового кодування супроводжується деякими обмеженнями. По-перше, це значна обчислювальна складність процедур відновлення даних: у разі відмови збільшується час затримки до доступу, що в певних умовах є критичним, адже є сервіси, де є чіткі вимоги до затримки отримання інформації.

Окремо необхідно виділити проблеми, що характерні для промисловості, оскільки сучасні індустріальні системи включають у себе сенсорні та мережі керування, які функціонують під впливом складних електромагнітних умов, збереження даних у хмару, аналітика даних та використання штучного інтелекту для цього.

Промислові підприємства, енергетичні установки та виробничі лінії створюють середовище з інтенсивними імпульсними завадами, що безпосередньо впливає на надійність передавання даних. Базуючись на результатах наукових досліджень, у таких умовах помилки часто мають груповий характер і проявляються у вигляді втрати або спотворення цілих фрагментів інформації [5, 6].

Для вирішення цих проблем науковці почали шукати способи їх розв’язання. Пошуки привели їх до непозиційної системи числення, а саме до системи залишкових класів. Ключова особливість системи залишкових класів полягає в тому, що кожен модуль працює сам по собі, тому помилка в одному місці не зіпсує всі обчислення. Але в існуючих методах декодування в СЗК можна виділити основну проблематику – високі обчислювальні витрати, особливо якщо помилки не поодинокі, а мають масовий характер. Саме ця проблема обмежує варіанти використання СЗК, адже в високонавантажених системах часто є багато помилок, що понесе за собою великі затримки у відновленні інформації за рахунок методів відновлення СЗК.

1.2 Теоретичні основи системи залишкових класів (СЗК)

Система залишкових класів – це непозиційна система числення, архітектура якої базується на фундаментальних положеннях теорії чисел і дозволяє кардинально змінити підхід до комп’ютерної арифметики. Якщо в звичайних системах числення кожна цифра має свою вагу залежно від позиції і все обчислюється послідовно, СЗК використовує принцип модулярності. Ця особливість дозволяє реалізувати табличну арифметику, що суттєво підвищує швидкодію обчислень завдяки заміні послідовної обробки на паралельну [7].

Математично СЗК визначається впорядкованим набором взаємно простих цілих чисел $m_1, m_2, \ldots, m_n$, які називаються модулями системи. Щоб система працювала правильно, модулі обов'язково мають бути взаємно простими, інакше виникнуть неоднозначності, що і показано в роботах [8], [9]. Динамічний діапазон системи M, в межах якого будь-яке ціле число X може бути кодоване, визначається як добуток усіх модулів. Будь-яке ціле число в СЗК представляється у вигляді вектора, що показано на рисунку 1.2.

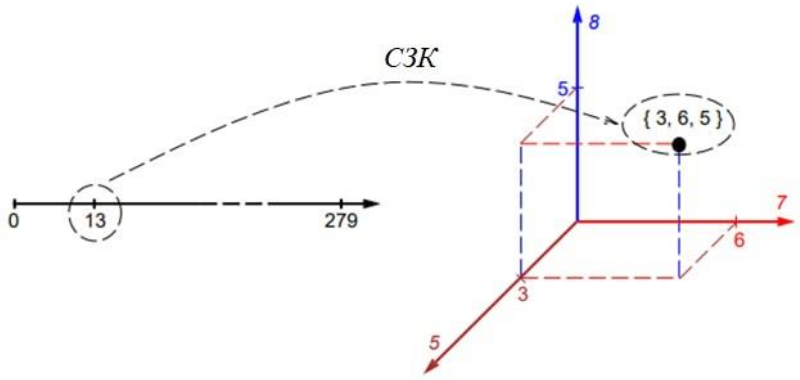


Рисунок 1.2 – Приклад переносу числа в СЗК [10]

Ключовою особливістю представлення чисел у системі залишкових класів є вбудований паралелізм виконання арифметичних операцій. На відміну від позиційних систем числення, у яких обчислення супроводжуються послідовними міжрозрядними переносами, СЗК дозволяє виконувати базові арифметичні дії незалежно для кожного модуля. Операції додавання, віднімання та множення реалізуються помодульно, без взаємного впливу між окремими обчислювальними каналами [10]. Формально результат арифметичної операції над числами X і Y у СЗК визначається набором залишків, кожен з яких обчислюється автономно за відповідним модулем. Такий підхід прибирає необхідність формування ланцюгів переносу, які є основним обмежувальним фактором швидкодії у двійкових суматорах великої розрядності. Так як немає необхідності в формуванні ланцюгів переносу, це суттєво покращує продуктивність обчислень, особливо в задачах з великою кількістю арифметичних операцій.

Практичні переваги модулярного паралелізму детально проаналізовано у роботі [11]. Запропоновані авторами методи оптимізації операції додавання ґрунтуються на розподілі обчислень між незалежними каналами, що дає змогу застосовувати суматори малої розрядності. Така архітектура не лише спрощує апаратну реалізацію, а й дозволяє зменшити енергоспоживання обчислювальних вузлів. Окрім цього, додатковою перевагою є підвищення надійності системи, оскільки помилка в одному з модулів не поширюється на інші залишки, створюючи основу для ефективних механізмів контролю та виявлення збоїв [12].

Але в системі залишкових класів є і свої проблеми, пов’язані зі специфікою виконання немодульних операцій. В класичних монографіях, присвячених системі залишкових класів, операції ділення, порівняння чисел, визначення знака або перевірки переповнення, потребують знання повного значення числа, що у СЗК безпосередньо недоступне [13]. Для їх реалізації часто застосовуються процедури зворотного перетворення в позиційну систему числення, що в свою чергу несе збільшення затримок для реалізації цих перетворень, що зменшує ефективність системи у разі частого використання даних операцій [7].

Отже, підсумовуючи переваги та недоліки системи залишкових класів, можна сказати, що в СЗК є потенціал для вирішення задач з збереження та відновлення інформації, зокрема в хмарних середовищах, які потребують підвищеного рівня надійності зберігання даних. Але головною проблемою класичної СЗК є великі затримки при обробці великої кількості помилок, а тому для вирішення таких проблем була розроблена Надлишкова Система Залишкових Класів.

1.3 Принципи побудови надлишкової системи залишкових класів (НСЗК)

Класична система залишкових класів, що складається лише з інформаційних модулів, орієнтована передусім на досягнення максимальної швидкодії обчислень. Але сама структура СЗК не передбачає механізмів, що зможуть контролювати коректність відновлених даних. У межах робочого діапазону кожному можливому

набору залишків $(x_1,x_2,...,x_k)$ відповідає певне допустиме числове значення, що унеможливлює пряме виявлення спотворень.

За відсутності механізмів контролю будь-яке випадкове або навмисне порушення одного з залишків, спричинене апаратним збоєм чи впливом зовнішніх факторів, призводить до перетворення вихідного правильного числа на інше коректне з точки зору системи значення хибного числа. Оскільки обидва числа належать допустимому діапазону, система просто не здатна зафіксувати свою помилку, адже немає додаткової інформації для перевірки точності відновлених даних. Саме через цю причину СЗК не розглядають при створенні систем з підвищеним рівнем збереження та відновлення даних.

Саме для вирішення цієї проблеми СЗК було прийнято рішення розширити базис, адже тоді з'являється можливість перевіряти достовірність відновленої інформації. У результаті формується НСЗК, яка за рахунок надлишковості отримує модифікований механізм виявлення та виправлення помилок, в якому відновлена інформація не спотворюється, адже є додаткові дані для перевірки достовірності [14].

Теоретичні основи побудови надлишкових систем залишкових класів спираються на класичні положення, закладені в роботах [15] та [16]. Формування надлишковості здійснюється шляхом доповнення набору з k інформаційних модулів $m_1,m_2,...,m_k$ додатковими r контрольними модулями $m_{k+1},...,m_{k+r}$. Така структура створює умови для розмежування допустимих і недопустимих комбінацій залишків, що є базою для реалізації ефективних алгоритмів контролю та виправлення помилок. При цьому зберігається фундаментальна вимога: усі модулі розширеного набору повинні бути попарно взаємно простими [10].

Суть методу полягає в наступному: якщо в результаті збою один або декілька залишків x_i спотворюються, то виправлене за повним базисом число X з високою ймовірністю виходить за межі інформаційного діапазону M, потрапляючи в нелегітимний діапазон. Тобто умова $X<M$ порушується. Саме завдяки цьому принципу в НСЗК виявляються помилки [17, 18]. Певні наукові роботи пропонують ефективну схему виявлення такого виходу за межі діапазону шляхом оцінки

величини числа [19, 20]. Використання змішаної системи числення дозволяє однозначно визначити належність числа до легітимного діапазону без необхідності повного відновлення його значення. Аналогічний підхід підтримується у дослідженні [21], де показано, що ефективність виявлення залежить від величини надлишкового добутку M_R: чим він більший, тим менша ймовірність того, що спотворене число випадково залишиться в межах діапазону.

Основним принципом завадостійкості НСЗК є кодова відстань. У теорії кодування в СЗК використовується метрика Геммінга, яка визначається як кількість позицій, у яких відрізняються вектори двох чисел. НСЗК з r контрольними модулями можна розглядати як (n,k) код з максимальною рознесеною відстанню [22-24].

Також необхідно виділити і таку особливість НСЗК: наявні класичні методи ефективно себе показують лише в корекції одиничних помилок (при r=2) або подвійних помилок (при r=4) [25-27]. Але сучасні канали передачі та зберігання даних мають настільки велику кількість інформації, що одиничних або подвійних помилок або немає або мало. В основному це помилки високої кратності, коли кількість пошкоджених модулів перевищує межу ⌊r/2⌋.

1.4 Метод проекцій на основі Китайської теореми про залишки

Теоретичним підґрунтям перетворення чисел із непозиційної системи залишкових класів у звичну позиційну форму є Китайська теорема про залишки. Саме вона дозволяє пов’язати набір окремих залишків із єдиним цілим числом. У дослідженнях, присвячених завадостійкому кодуванню, підкреслюється, що КТПЗ фактично виконує роль математичного інструменту реконструкції початкового значення [28]. Маючи набір залишків $(x_1,x_2,\ldots,x_N)$ отриманих за відповідними модулями $m_1,m_2,\ldots,m_N$, можна однозначно відновити число X. Ключовою умовою тут є взаємна простота модулів – лише за цієї вимоги гарантується коректність і єдиність результату в межах заданого діапазону [29]. Згідно з КТПЗ, позиційне

значення числа обчислюється як сума добутків кожного залишку на відповідну ортогональну вагу базису, взята за модулем повного динамічного діапазону.

Алгоритмічна реалізація методу проекцій, детально описана у дослідженні [30], базується на ідеї поетапного виключення модулів, які потенційно можуть містити помилки. Інакше кажучи, система перевіряє різні варіанти відновлення числа, тимчасово ігноруючи окремі компоненти базису. Числовий результат, отриманий після такого виключення, називають проекцією та позначають X_{-i}. Він визначається за допомогою КТПЗ, але вже на основі зменшеного набору модулів, з якого вилучено підозрілий залишок або їх групу.

Якщо припустити, що в системі виникла помилка кратності t, алгоритм передбачає виключення саме t модулів із загального набору та виконання відновлення числа за рештою $N - t$ модулів. Після цього здійснюється перевірка отриманого результату. Коли виконується умова $X_{-i} < M$, тобто відновлене значення потрапляє в межі робочого діапазону системи, робиться висновок, що вилучені модулі дійсно містили помилки. Водночас залишки, які брали участь у відновленні, вважаються достовірними. Отримане число X_{-i} у такому випадку приймається як правильне початкове значення, що означає успішне виявлення та корекцію помилки. Візуально принцип роботи методу проекцій на основі китайської теореми про залишки зображено на рисунку 1.3.

Хоча метод проекцій має чітке математичне підґрунтя і демонструє гарантовану ефективність при локалізації помилок невеликої кратності, детальний аналіз його роботи виявляє серйозні обчислювальні обмеження, які стають особливо помітними під час масштабування системи.

У наукових дослідженнях, зокрема в роботі [31], наголошується, що кожен крок алгоритму супроводжується операціями множення та додавання над числами, величина яких перевищує робочий динамічний діапазон системи. На практиці це означає необхідність застосування багаторозрядних модульних суматорів. Подібні апаратні вимоги фактично нівелюють одну з ключових переваг систем залишкових класів – можливість ефективного паралельного виконання обчислень [32].

Рисунок 1.3 – Методу проекцій на основі китайської теореми про залишки в надлишковій системі залишкових класів

Ще складнішою ситуація стає тоді, коли виникають помилки підвищеної кратності, тобто коли $t > 1$. У такому випадку алгоритм стикається з явищем, яке в теорії обчислювальних методів часто називають комбінаторним вибухом. Його сутність полягає в різкому зростанні кількості можливих комбінацій модулів, які необхідно перевірити для пошуку помилкових компонентів. Число проекцій, що підлягають обчисленню та подальшій перевірці, визначається біноміальним коефіцієнтом.

Наведемо просту ілюстрацію цієї залежності. Для системи, що містить п’ять модулів, локалізація одиничної помилки потребує не більше ніж п’яти ітерацій алгоритму. Однак ситуація кардинально змінюється у випадку складніших конфігурацій. Наприклад, якщо в промисловій системі з п’ятнадцяти модулів виникають чотири пакетні помилки, алгоритму доведеться послідовно сформувати й перевірити вже 1365 окремих проекцій.

Таким чином, обчислювальна складність методу зростає відповідно до комбінаторної залежності. Для систем цифрової обробки сигналів, що працюють у режимі реального часу, подібне навантаження виявляється практично неприйнятним. Саме ця обставина підкреслює актуальність пошуку

альтернативних підходів – алгоритмів, здатних виявляти множинні помилки без необхідності перебору великої кількості комбінацій.

1.5 Синдромні методи декодування в НСЗК

Іншим підходом до задачі виявлення та виправлення помилок у надлишкових системах залишкових класів є метод синдромного декодування. Його концептуальна основа походить із класичної лінійної теорії алгебраїчних кодів, де синдроми широко використовуються для діагностики та локалізації помилок [33, 34]. Як підкреслено у роботі [35], на відміну від методу проекцій, який передбачає багаторазове відновлення позиційного представлення числа для різних комбінацій модулів, синдромні алгоритми працюють у значно компактнішому обчислювальному просторі. Цим простором виступає підмножина контрольних, тобто надлишкових модулів, що дозволяє істотно зменшити обсяг необхідних операцій.

Ключова ідея цього підходу полягає у формуванні спеціального діагностичного вектора – синдрому. Для коректного, тобто безпомилкового числа, значення такого вектора дорівнює нулю. Якщо ж у системі виникає збій, синдром набуває ненульового значення, сигналізуючи про наявність помилки.

На практиці формування синдрому найчастіше реалізується за допомогою процедури базисного розширення. Математична суть цієї процедури полягає у відновленні числа виключно за інформаційними залишками, після чого обчислюються його відповідні значення у контрольних модулях.

Відповідно до теореми, доведеної у [36], кожному можливому вектору помилки, кратність якого не перевищує корекційної здатності коду $t \leq \lfloor r/2 \rfloor$, відповідає строго визначений і єдиний синдром. Саме ця властивість робить можливим використання таблиць пошуку, за допомогою яких процедура корекції може виконуватися практично миттєво – достатньо лише зіставити отриманий синдром із заздалегідь підготовленою таблицею.

Очевидною перевагою синдрому як інструменту локалізації помилок є відсутність потреби у складних багаторозрядних арифметичних операціях [37]. На етапі діагностики помилки замість трудомістких модулярних обчислень використовується значно простіший механізм – адресація до пам’яті. Таким чином, важкі обчислювальні блоки можуть бути замінені швидким доступом до таблиці відповідностей.

Разом із тим, як показано у дослідженні [38], при масштабуванні системи цей підхід стикається з серйозними обмеженнями. Основна проблема пов’язана зі швидким зростанням розміру таблиці пошуку. Кількість її комірок визначається числом усіх можливих синдромів, які можуть виникнути для різних комбінацій помилок [39].

У найпростішому випадку, коли система повинна виправляти лише одну помилку, для НСЗК з N модулями необхідно зберігати невелику кількість записів. Для сучасних мікросхем пам’яті такий обсяг даних не становить проблеми. Однак ситуація кардинально змінюється, якщо система повинна працювати з помилками більшої кратності $t > 1$. Кількість необхідних записів стає пропорційною числу можливих поєднань спотворених модулів і різко збільшується зі зростанням параметрів системи. Візуалізацію синдромних методів зображено на рисунку 1.4.

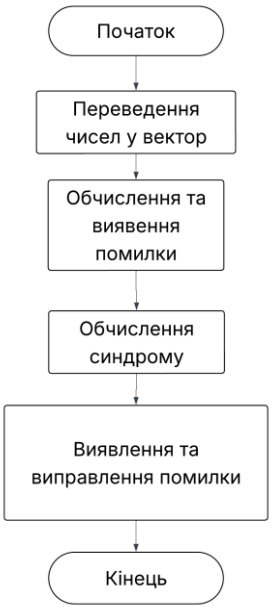


Рисунок 1.4 – Синдромні методи декодування в НСЗК

Отже, класичні табличні методи синдромного декодування мають суттєві обмеження масштабованості. Це фактично підтверджує необхідність пошуку нових підходів – аналітичних алгоритмів локалізації збоїв, які не залежать від великих таблиць і дозволяють уникнути комбінаторного вибуху як у часовому, так і в апаратному аспекті.

1.6. Методи контролю на основі системи зі змішаною основою

Альтернативою до традиційних методів, що вимагають роботи з величезними числами, є перехід на позиційну систему числення зі змішаною основою. Теоретичні засади цього підходу закладено у [40], а їх подальше формалізоване викладення міститься у класичній праці [41]. На відміну від непозиційних СЗК, де кожен модуль розглядається ізольовано, поліадичний код системи зі змішаною основою має строго визначену вагову структуру. Це дозволяє безпосередньо порівнювати числа за величиною і ефективно контролювати випадки переповнення робочого діапазону [42].

Як підкреслюють у [43], у надлишковій системі залишкових класів, де перші n модулів інформаційні, а наступні r – контрольні, будь-яке легітимне число має нульові коефіцієнти старших розрядів, що відповідають контрольним модулям.

Попри те, що система зі змішаною основою дозволяє уникнути арифметичних операцій над числами величини повного динамічного діапазону, її апаратна реалізація виявляється вкрай неефективною. Дослідження [44] демонструє, що рекурентна залежність між цифрами змішаної основи створює критичний шлях затримки довжиною *O(N)*, де *N* – загальна кількість модулів.

На відміну від КТПЗ, де N проекцій можна обчислити паралельно за O(1) модулярних кроків, у системі зі змішаною основою кожен наступний крок очікує завершення попереднього. У випадку пакетних помилок високої кратності t > 1, алгоритмічний процес ускладнюється катастрофічно: для локалізації t помилок доводиться перебирати C_N^{N-t} комбінацій модулів, а для кожної комбінації виконувати послідовний алгоритм Гарнера довжиною $N - t$ кроків[45, 46].

Цей ефект, який можна охарактеризувати як алгоритмічний колапс, робить систему зі змішаною основою абсолютно непридатним для виправлення помилок високої кратності у високошвидкісних системах передачі та обробки даних. Він підтверджує необхідність розробки принципово нових методів, які не залежать від послідовних обчислень і дозволяють уникнути комбінаторного вибуху як у часовій, так і в апаратній площині.

1.7 Наближені та модифіковані алгоритми контролю

Усвідомлення фундаментальних апаратних і часових обмежень традиційних методів декодування стало важливим стимулом для пошуку альтернативних математичних підходів, здатних оцінювати коректність представлення чисел у системі залишкових класів без виконання повнорозрядних модулярних операцій. Одним із таких підходів є концепція так званої «функції ядра», детально досліджена та формалізована у роботі [47], що потім отримала свій розвиток у роботі [48].

Сутність цієї концепції полягає у введенні спеціальної позиційної характеристики непозиційного коду, яка відображає множину залишків у значно компактніший числовий простір. При цьому зберігається властивість монотонності відображення, що дозволяє виявляти ситуації виходу числа за межі допустимого діапазону.

Однак як показано у ряді досліджень [49, 50], практична реалізація цього підходу пов’язана з низкою серйозних труднощів. Найбільш складною задачею є визначення оптимальних значень вагових коефіцієнтів для функції ядра, що фактично перетворюється на нелінійну оптимізаційну проблему високої розмірності. Крім того, у випадку виникнення помилок великої кратності процедура відновлення даних усе одно потребує перебору різних проекцій, тому фундаментальна проблема комбінаторної складності залишається невирішеною. Функція ядра лише пришвидшує перевірку окремої проекції, але не усуває необхідності їх множинного аналізу.

Інший, більш сучасний підхід, який активно застосовується на практиці, базується на використанні наближених обчислень, що спираються на Китайську теорему про залишки. Такий метод часто називають приблизним КТПЗ або фракційним КТПЗ. Його математичне обґрунтування наведено у роботі [51]. Основна ідея полягає у переході від операцій над великими цілими числами до обчислень із дробовими величинами у фіксованій точності.

Головна перевага цього підходу з апаратної точки зору полягає у суттєвому спрощенні арифметики [52]. Замість роботи з модулярними суматорами великої розрядності достатньо виконувати наближене додавання дробових компонентів, зберігаючи лише обмежену кількість старших бітів після коми. У результаті апаратна реалізація може використовувати стандартні двійкові суматори з розрядністю приблизно 12–16 бітів, що значно зменшує як площу кристала, так і затримку поширення переносу.

Практичне застосування цієї математичної моделі призвело до створення архітектур швидкого неітеративного декодування, запропонованих, зокрема, у [53]. Попри значний приріст швидкодії, який дозволяє наблизити продуктивність систем залишкових класів до швидкості традиційних позиційних арифметичних схем, такі алгоритми мають принципову ваду – похибку усічення.

Відкидання молодших бітів дробової частини неминуче породжує додаткову шумову складову. У ситуаціях, коли спотворене число розташоване поблизу межі допустимого діапазону, ця похибка може призвести до некоректного рішення компаратора.

У результаті виникають два небажані сценарії: система може або відхилити коректну проекцію, або, що ще критичніше, прийняти спотворене значення за правильне. Щоб зменшити ймовірність таких ситуацій, доводиться збільшувати кількість збережених дробових бітів, що частково нівелює первинні апаратні переваги методу.

Суттєвим недоліком усіх подібних підходів залишається їхня залежність від класичної стратегії перебору. Незважаючи на те, що використання функції ядра або

наближеного варіанту КТПЗ значно прискорює виконання окремої перевірки, це не змінює асимптотичної складності процесу пошуку помилкових каналів.

У випадку виникнення пакетної помилки з вагою $t > \lfloor r/2 \rfloor$ алгоритм усе одно змушений сформувати C_N^{N-t} різних проекцій і виконати для кожної з них відповідні обчислення – дробову суму або значення функції ядра. Як показано у дослідженнях [54],[55], навіть для відносно помірної конфігурації N = 15 та t =4 необхідність виконання 1365 операцій додавання створює неприйнятну затримку для систем потокової обробки даних і телекомунікаційних протоколів.

Таким чином, локальні оптимізації внутрішніх математичних операцій – незалежно від того, чи застосовується функція ядра, чи дробова реалізація КТПЗ – не здатні усунути фундаментальну проблему комбінаторного зростання складності. Це свідчить про необхідність пошуку принципово нових математичних моделей, які дозволять локалізувати множинні помилки за поліноміальний час.

1.8 Проблема помилок високої кратності в інформаційних системах

Зменшення розмірів транзисторів і перехід до нанометрових норм проєктування повністю змінило характер помилок у цифрових пристроях. Якщо раніше переважали поодинокі незалежні збої [56], то сучасні інтегральні схеми з високою щільністю елементів дедалі частіше демонструють множинні порушення.

Зменшення геометричних розмірів і робочої напруги призводить до того, що одна високоенергетична частинка здатна індукувати зміну стану одразу кількох сусідніх комірок пам’яті. Такі зміни особливо небезпечні для систем залишкових класів, тому що локальне фізичне ушкодження кристала може одночасно спотворити декілька модулів, що, в свою чергу, впливає на коректність відновлення числа. Основним підходом для покращення відмовостійкості є використання апаратної надлишковості, а саме потрійного модульного резервування. Принцип його роботи полягає у паралельному дублюванні функціональних блоків із подальшим голосуванням результатів.

Проте реалізація потрійного модульного резервування потребує більш ніж дворазового збільшення апаратних ресурсів і суттєво підвищує енергоспоживання [57-59]. Це є критичним для вбудованих платформ, ПЛІС та пристроїв IoT. Унаслідок цього акцент зміщується в бік алгоритмічної надлишковості, яку забезпечують коди в системі залишкових класів [60].

При передачі даних бездротовим шляхом ми отримуємо нові проблеми. В умовах інтенсивних електромагнітних завад помилки часто мають пакетний характер. У межах НСЗК це означає, що кількість спотворених залишків t може перевищувати класичну межу виправлення ⌊r/2⌋. Коли $t>\lfloor r/2 \rfloor$, стандартні процедури декодування втрачають гарантовану коректність, а ефективність відновлення різко знижується [61].

Проаналізувавши, ми маємо такі висновки – що за наявності множинних помилок алгоритми, засновані на Китайській теоремі про залишки, стикаються з явищем «комбінаторного вибуху». Щоб ідентифікувати t пошкоджених модулів, необхідно перебрати велику кількість можливих підмножин залишків у пошуках такої комбінації, яка відновлює число в допустимому діапазоні. Зі зростанням кратності помилки обчислювальні витрати збільшуються експоненційно, що призводить до неприйнятних затримок у системах реального часу. Кількість ітерацій перевірок до кількості помилок зображена нижче на рисунку 1.5.

Як вже було згадано, сучасні мережі та системи потребують високу надійність та мінімальні затримки. Для реалізації режиму наднадійного зв’язку з малою затримкою (URLLC) необхідні механізми, здатні коригувати множинні помилки практично миттєво.

Існуючі реалізації НСЗК, що застосовують послідовні алгоритми декодування на основі MRS, не забезпечують потрібної швидкодії при значній кількості помилок, що, в свою чергу, не відповідає сучасним вимогам.

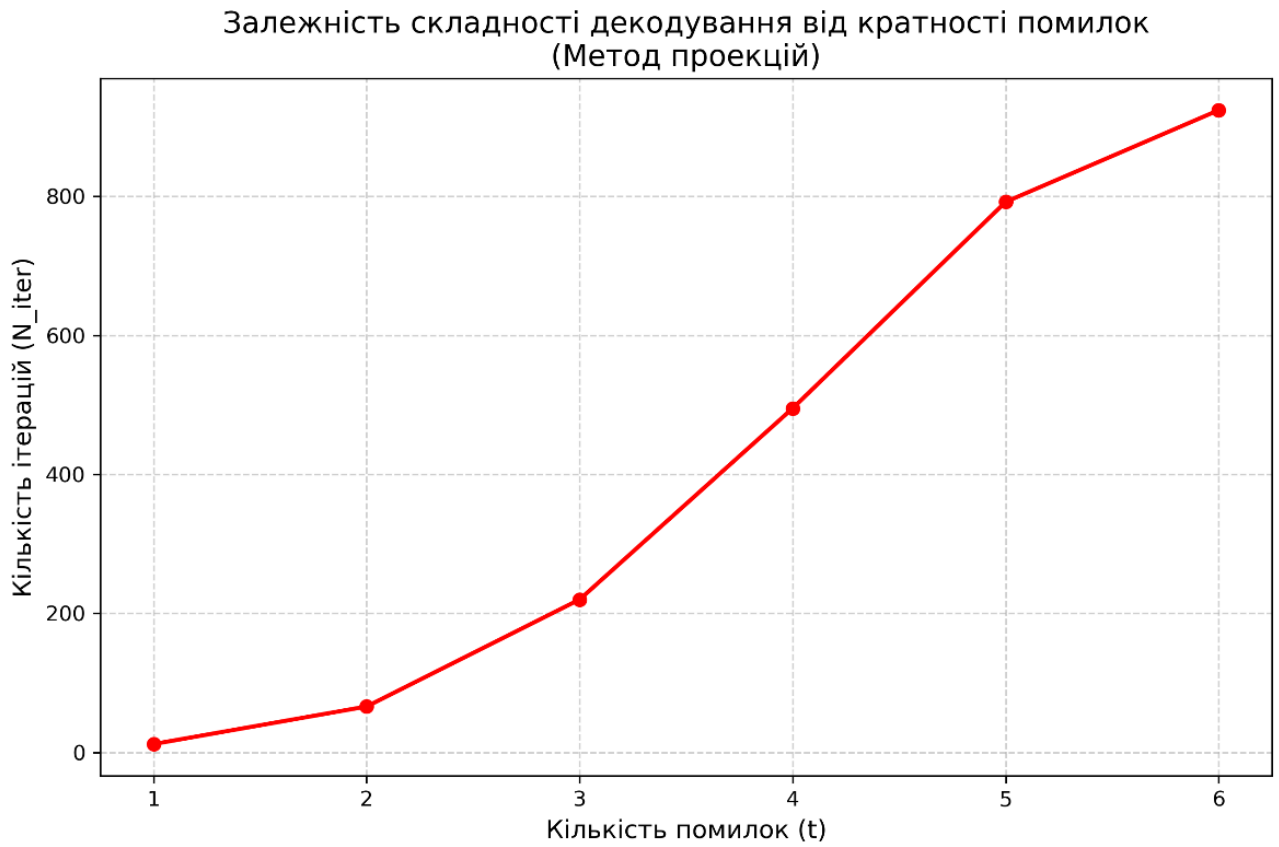


Рисунок 1.5 – Графік залежності кількості ітерацій до кількості помилок в методі проекцій

1.9 Висновки та постановка задачі

В першому розділі проведено огляд розвитку сучасних технологій та їх потреб, як намагались задовольнити ці потреби. З збільшенням потреб та появою нових вимог – з’явилась потреба в нових рішеннях і їх знайшли спочатку в система залишкових класів а згодом і в надлишковій системі залишкових класів.

Вбудовану в СЗК можливість проводити операції паралельно а також виправляти поодинокі помилки зацікавила науковців, але з приходом нових технологій, збільшенням розмірів інформації, що в свою чергу збільшує кількість помилок, зменшенням розмірів транзисторів і робочих напруг, що змінили природу апаратних помилок вчені прийшли до висновку що СЗК не є ефективним рішенням.

Так вчені прийшли до створення надлишкової системи залишкових класів або ж НСЗК. Оскільки простим дублюванням інформації вже не обійтись, бо це несе за собою великі грошові витрати, класична система залишкових класів погано

себе показує при множинних помилках а виправити це апаратно також дуже затратно, було прийнято рішення створити математичне рішення, а саме алгоритмічну надлишковість.

Існуючі методи та алгоритми для виявлення та виправлення помилок не є ефективними. Алгоритми на основі китайської теореми про залишки числення показують задовільні результати лише при малій кількості помилок, при їх збільшенні виявлення та виправлення викликають значні затримки.

Методи на основі змішаної системи числення мають наступні недоліки: послідовний характер обробки та складна логіка прийняття рішень спричиняють додаткові затримки, що негативно впливає на швидкодію.

Якщо було виявлено, що помилок більше ніж половина наявних контрольних модулів то звичайні методи перебору проекцій зіштовхуються з експоненційним зростанням складності та затримок, що в свою чергу виключає їх з переліку можливих варіантів вирішення проблем великих затримок.

Сучасні системи вимагають рішень, які поєднують високу швидкодію, помірні апаратні витрати та здатність працювати за умов значної кратності збоїв, а наявні методи та алгоритми не можуть все це забезпечити.

Необхідно покращити один з наявних методів, для цього нам потрібно:

- проаналізувати відомі методи виявлення та корекції помилок у системах залишкових класів та виявити їхні обмеження;
- розробити математичну модель процесу виявлення помилок високої кратності на основі надлишкової системи залишкових класів;
- удосконалити метод виявлення помилок високої кратності на основі надлишкової системи залишкових класів для усунення ефекту алгоритмічного колапсу;
- здійснити програмну реалізацію та експериментальне дослідження розробленого методу виявлення помилок високої кратності.

2 МОДЕЛЮВАННЯ ПРОЦЕСУ ВИЯВЛЕННЯ ПОМИЛОК ВИСОКОЇ КРАТНОСТІ НА ОСНОВІ НАДЛИШКОВОЇ СИСТЕМИ ЗАЛИШКОВИХ КЛАСІВ

2.1 Функціональні та архітектурні компоненти моделі процесу виявлення помилок високої кратності

Модель системи контролю даних не є єдиним монолітним алгоритмом, а являє собою складний конвеєр, який складається з послідовності взаємопов’язаних функціональних блоків. Всього є 5 блоків, а саме:

1. Блок формування базису та прямого перетворення даних.
2. Блок передачі та обробки.
3. Блок агрегації та первинного аналізу даних.
4. Блок математичного перетворення та локалізації збоїв.
5. Блок верифікації та прийняття рішень.

Кожен із функціональних блоків системи виконує чітко визначену роль – від етапу первинного кодування до формування остаточного рішення щодо наявності помилок. Лише розуміння того, як саме ці компоненти взаємодіють між собою, дає змогу коректно описати їх у межах математичної моделі та, що не менш важливо, виявити слабкі місця, які проявляються при появі помилок високої кратності.

Початковою ланкою є блок формування базису та прямого перетворення даних. Його функція полягає у створенні структурної надлишковості, без якої неможливе подальше виявлення збоїв. На вході цей блок отримує інформацію у звичному двійковому представленні, після чого перетворює її у набір незалежних залишків. Саме на цьому етапі закладається фундаментальний поділ системи на інформаційні та контрольні канали. Іншими словами, дані не просто розподіляються по основних обчислювальних трактах, а доповнюються надлишковими компонентами. Від правильності роботи цього вузла залежить узгодженість усього подальшого процесу обробки.

Саме на цьому етапі архітектурно закладається поділ системи на інформаційну та контрольну частини. Блок прямого перетворення відповідає за те,

щоб початкові дані були представлені не лише в основних робочих каналах, а й згенерували відповідні значення для додаткових, надлишкових каналів. Коректна робота цього компонента гарантує, що вся подальша система отримає математично узгоджений вектор даних, готовий до безпечної передачі або обробки.

Наступний етап пов’язаний із блоком передачі та обробки даних, який є ключовим з точки зору виникнення помилок. У реальних умовах його роль можуть виконувати різні елементи: шини даних, обчислювальні модулі, оперативна пам’ять або навіть канали зв’язку. У межах моделі цей блок розглядається як джерело потенційних спотворень. Саме тут відбувається вплив на залишки – як фізичний (наприклад, через апаратні дефекти), так і логічний. Для моделювання складних сценаріїв він налаштовується таким чином, щоб імітувати пакетні помилки, тобто одночасне пошкодження кількох каналів. У результаті змінюються значення одразу декількох залишків, що створює найбільш складні умови для подальшого аналізу.

Після проходження через середовище з завадами дані надходять до блоку агрегації та первинної обробки. Його основне завдання – зібрати всі залишки в єдиний вектор, незалежно від того, чи є вони інформаційними, чи контрольними. Цей блок фактично виконує роль точки збору та синхронізації. Він не намагається одразу виправляти помилки, оскільки на цьому етапі ще відсутня інформація про те, які саме канали зазнали спотворення. Його функція обмежується фіксацією поточного стану системи та передачею сформованого вектора до наступного етапу обробки.

Центральне місце у всій архітектурі займає блок математичного перетворення та локалізації помилок. Саме він виконує основну інтелектуальну роботу, застосовуючи спеціалізовані алгоритми декодування для аналізу отриманих даних. У цьому блоці система або намагається відновити глобальне значення числа, або оцінює його характеристики на основі наявних залишків. Водночас саме тут виникають найбільші труднощі у випадку помилок високої кратності. Коли кількість пошкоджених каналів значна, класичні алгоритми

змушені перебирати велику кількість можливих варіантів, що призводить до різкого зростання обчислювального навантаження і затримок у роботі системи.

Завершальним елементом є блок верифікації та прийняття рішень. Він аналізує результати, отримані на попередньому етапі, і порівнює їх із встановленими межами допустимого діапазону. Фактично цей вузол відповідає за винесення остаточного вердикту щодо коректності даних. Якщо відновлене значення виходить за межі легітимного простору, формується сигнал про наявність помилки.

Однак саме на цьому етапі може проявитися одна з найнебезпечніших проблем – ефект аліасингу. У ситуаціях із множинними збоями сильно спотворені дані інколи можуть випадково відповідати допустимому діапазону, що призводить до помилкового прийняття їх як коректних. Таким чином, надійність усієї системи визначається не лише ефективністю окремих алгоритмів, а й узгодженістю роботи всіх п’яти блоків, кожен із яких впливає на кінцевий результат.

2.2 Модель функціональних блоків моделі виявлення помилок

Опишемо кожен з наявних п’яти блоків математично, для кращого розуміння їх роботи та даних, якими вони оперують.

2.2.1 Модель блоку формування базису та прямого перетворення даних

Нехай на вхід цього блоку надходить ціле позитивне число, яке несе корисну інформацію. Позначимо це число великою літерою X. Блок містить у собі набір заздалегідь визначених модулів, які позначаються як:

$$m_1, m_2, \dots, m_N, \tag{2.1}$$

де m – заздалегіть визначений модуль;

N – загальна кількість модулів у системі.

Математична операція, яку виконує цей блок, описується формулою:

$$x_i = X\ (mod\ m_i), \tag{2.2}$$

де x_i – залишок для конкретного каналу зв'язку;

X – початкове інформаційне число;

m_i – значення конкретного модуля, на яке відбувається ділення.

Результатом роботи цього першого блоку є формування ідеального, безпомилкового масиву даних, який математично записується як вектор залишків:

$$x_1, x_2, \dots, x_N. \tag{2.3}$$

2.2.2 Модель блоку передачі та обробки

Під час проходження ідеального залишку x_i через фізичне середовище, до нього випадковим чином додається певне ціле число, яке моделює апаратний збій. Це хибне значення позначено малою літерою *e*. Відповідно, на виході з цього середовища формується нове, потенційно спотворене значення, яке позначено літерою *z*. Математичне рівняння, що описує роботу цього блоку-генератора завад, має вигляд:

$$z_i = (x_i + e_i)(mod\ m_i), \tag{2.4}$$

де z_i – спотворений залишок, що надійшов на приймач;

x_i – це початковий правильний залишок;

e_i – позначає саму величину внесеної помилки, а оператор ділення з остачею.

Сукупність усіх внесених помилок формує вектор помилки, який позначається великою літерою *E*. Цей вектор записується як:

$$E = e_1, e_2, \dots, e_N. \tag{2.4}$$

Якщо завада була потужною і вразила відразу багато каналів, кількість ненульових значень е у цьому векторі буде великою. Ця кількість позначається літерою *t* і називається кратністю помилки, що є ключовим параметром для нашої моделі.

2.2.3 Модель блоку агрегації та первинного аналізу даних

Його математичне завдання є найпростішим з точки зору обчислень, але критично важливим для структуризації даних. Він збирає всі розрізнені значення z_i, які надійшли з різних апаратних каналів, і об'єднує їх у єдину математичну структуру – вектор прийнятих залишків. Цей вектор позначається великою літерою *Z* і записується як:

$$Z = z_1, z_2, \dots, z_N, \tag{2.5}$$

де Z – весь масив інформації, що дійшов до декодера;

z_1, z_2 – індивідуальні значення, прийняті по кожному окремому модулю. Важливість цього блоку полягає в тому, що для всієї подальшої системи єдиним джерелом вхідних даних є виключно вектор *Z*. Система на цьому етапі не має жодної інформації ні про початкове число *X*, ні про згенерований вектор помилки *E*, і змушена працювати лише з тим масивом *Z*, який передав блок агрегації.

2.2.4 Модель математичного перетворення та локалізації збоїв

Четвертий блок відповідає за математичне перетворення та спробу локалізації збоїв. Саме сюди надходить агрегований вектор *Z*. Математично роботу цього інтелектуального блоку можна описати через функцію перетворення, яку ми позначимо великою літерою *F*. Рівняння роботи цього блоку має узагальнений вигляд:

$$Y = F(Z), \tag{2.6}$$

де Z – вхідний вектор прийнятих залишків;

F – використаний математичний алгоритм, наприклад метод проекцій на основі китайської теореми про залишки або обчислення коефіцієнтів змішоної основи;

Y – глобальне числове значення або набір діагностичних ознак, які алгоритм зміг відновити.

Проблема цього блоку полягає в тому, що коли кратність помилки t є високою, функція F стає надзвичайно важкою для обчислення. Замість одного простого прорахунку, блоку доводиться застосовувати функцію F до різних комбінацій елементів вектора Z, що призводить до катастрофічного збільшення кількості математичних операцій та різкого гальмування всього процесу виявлення.

2.2.5 Модель блоку верифікації та прийняття рішень

Математичний апарат цього блоку базується на системі строгих нерівностей. Отримане з попереднього блоку відновлене значення Y порівнюється з робочим динамічним діапазоном системи. Цей діапазон позначається великою літерою M і обчислюється як добуток лише інформаційних модулів:

$$M = m_1 * m_2 * \ldots * m_n, \tag{2.7}$$

де $m_1, m_{2,\ldots,} m_n$ – інформаційні модулі.

Алгоритм прийняття рішень описується простою логічною умовою. Якщо відновлене значення строго менше за робочий діапазон, тобто виконується нерівність $Y < M$, блок робить математичний висновок, що помилок немає або вони були успішно скориговані. Якщо ж виконується умова, при якій відновлене значення дорівнює або ж більше ніж інформаційний добуток, блок фіксує факт

потрапляння числа в нелегітимну зону і генерує сигнал про виявлення апаратного збою.

Найбільша небезпека на цьому етапі виникає тоді, коли вектор помилки високої кратності E спотворює дані настільки сильно, що хибно відновлене число Y знову стає меншим за M. Цей математичний збій обманює п’ятий блок, змушуючи його прийняти пошкоджені дані за правильні, що вимагає розробки принципово нових моделей контролю.

2.3 Формалізація базових математичних параметрів та поширення помилок

Для того, щоб система виявлення помилок могла функціонувати математично коректно, необхідно чітко визначити кількісні параметри її архітектури. Базовим параметром будь-якої системи залишкових класів є загальна кількість апаратних каналів або модулів, яка визначає її пропускну здатність та рівень захищеності. Цей параметр описується найпростішим лінійним рівнянням:

$$N = n + r, \tag{2.8}$$

де N – загальна кількість усіх модулів, які використовуються в обчислювальній системі;

n – кількість виключно інформаційних модулів, головне завдання який полягає в зберіганні та обробці корисних даних користувача;

r – кількість додаткових, або ж контрольних модулів.

Контрольні модулі не несуть корисної інформації про саме число, а додаються в систему виключно як математичний баласт для пошуку, локалізації та виправлення апаратних збоїв. Саме значення параметра r визначатиме, наскільки потужним є захист нашої інформаційної системи.

На основі цього базового розділення модулів формується математична структура числового простору системи, яка описується трьома різними

динамічними діапазонами. Перший і головний з них – це робочий динамічний діапазон, який позначається великою літерою *M*, формула якого (2.7).

Усі цілі числа, які строго менші за значення *M*, утворюють легітимну зону системи. Другий простір – це надлишковий діапазон, який ми позначимо великими літерами *MR*. Він обчислюється як добуток усіх введених контрольних модулів:

$$MR = m_{n+1}, m_{n+2}, \dots, m_{n+r}. \quad (2.9)$$

Значення цього діапазону прямо пропорційне можливостям системи з виявлення помилок. Третій простір – це повний динамічний діапазон усієї системи, який позначається як M_{total}. Математично він визначається як добуток робочого та надлишкового діапазонів:

$$M_{total} = M * MR. \quad (2.10)$$

Всі числа, які більші або дорівнюють робочому діапазону *M*, але залишаються меншими за повний діапазон M_{total}, формують нелегітимну зону. Механізм виявлення збоїв базується на тому, що будь-яка апаратна помилка переміщує правильне число з дозволеної зони в заборонену.

Для того щоб кількісно оцінити рівень руйнівного впливу завад на систему, вводиться математичний параметр кратності помилки, який позначається малою літерою *t*. Як вже було сказано в описі роботи блоку агрегації та первинного аналізу даних, під час передачі даних формується вектор помилки *E*, який містить значення спотворень для кожного каналу. Параметр *t* вказує на точну кількість каналів, у яких значення помилки не дорівнює нулю.

Здатність математичної моделі протистояти цим спотворенням описується іншим критично важливим параметром – мінімальною кодовою відстанню, яка позначається великою літерою *D*. Цей параметр показує, наскільки далеко розташовані одне від одного правильні математичні значення в просторі системи. Формула для обчислення кодової відстані є максимально спрощеною і має вигляд:

$$D = r + 1, \tag{2.11}$$

де D – кодова відстань;

r – кількість контрольних модулів.

З цього рівняння стає очевидним, що кожен новий контрольний модуль лінійно збільшує кодову відстань, розширюючи можливості декодера.

Спираючись на значення кодової відстані, фундаментальна математична теорія дозволяє вивести строгу нерівність, яка визначає максимальну межу працездатності класичних алгоритмів корекції. Ця нерівність має вигляд:

$$t \leq r/2, \tag{2.12}$$

де t – фактична кількість пошкоджених модулів, тобто кратність помилки;

r – загальна кількість модулів у системі.

Ділення на 2 в цій нерівності вказує на те, що система здатна виправити лише половину від своєї надлишкової ємності. При цьому передбачається, що результат цього ділення завжди округлюється в меншу сторону до найближчого цілого числа. Тобто, при $r = 5$ можна виправляти лише поодинокі або подвійні помилки.

Проблема помилок високої кратності, яка є головним предметом нашого дослідження, виникає саме в той момент, коли ця фундаментальна нерівність порушується. Якщо потужна електромагнітна завада пошкоджує кількість каналів t, яка є більшою за значення $r/2$, система переходить у стан математичної невизначеності. Це явище в теорії інформації називається ефектом накладання або аліасингом. Суть явища полягає в тому, що вектор помилки високої кратності спотворює правильне число настільки сильно, що нове, хибне значення може випадково співпасти з іншим дозволеним числом з легітимного діапазону M, або ж вказувати на хибну конфігурацію помилок з меншою вагою.

У таких умовах будь-який класичний алгоритм виправлення помилок приймає неправильне математичне рішення, оскільки він завжди шукає вектор

помилки з найменшим значенням *t*. Саме тому моделювання і вивчення обмежень цієї базової нерівності є критично необхідними для переходу до наступного етапу – розгляду математичного апарату базових перетворень, який потрібно вдосконалити.

2.4 Математичний апарат базових перетворень

Для того, щоб інтелектуальний блок локалізації помилок міг перевірити, чи вийшло число за межі легітимного діапазону, йому необхідно перетворити набір незалежних залишків назад у єдине ціле позиційне число або отримати його позиційні характеристики. У математичній теорії систем залишкових класів існують два основних методи для виконання цього зворотного перетворення:

1. Китайська теорема про залишки.
2. Система числення зі змішаною основою.

Розглянемо перший з них. Китайська теорема про залишки, яку традиційно позначають абревіатурою CRT, дозволяє обчислити початкове ціле число за допомогою єдиної математичної операції зваженого додавання. Математична формула цього перетворення у максимально спрощеному вигляді записується так:

$$X = (x_1 * W_1 + x_2 * W_2 + \ldots + x_N * W_N)(mod\ M_{total}), \qquad (2.13)$$

де X – відновлене ціле позиційне число, яке є кінцевим результатом перетворення;

x_1, x_2, x_N – значення залишків у кожному відповідному інформаційному та контрольному каналі;

W_1, W_2, W_N – ортогональна вага конкретного каналу;

$(mod\ M_{total})$ – після додавання всіх добутків необхідно знайти остачу від ділення отриманої суми на повний динамічний діапазон системи.

Ортогональна вага W для кожного каналу є критично важливим компонентом цієї моделі. Математично вона обчислюється за окремою формулою:

$$W_i = M_i * w_i, \quad (2.14)$$

де M_i – добуток усіх модулів системи, за винятком поточного модуля m_i;

w_i – мультиплікативна інверсія, ціле число яке при множення на M_i дає результат, остача від ділення якого на поточний модуль m_i дорівнює 1.

Оскільки значення модулів системи обираються заздалегідь і не змінюються в процесі роботи комп'ютера, всі ортогональні ваги є абсолютними константами. У математичній моделі апаратного декодера це означає, що ці величезні числа не потрібно обчислювати щоразу; вони назавжди зберігаються в постійній пам'яті пристрою. Завдяки цьому процес відновлення числа в реальному часі зводиться лише до множення вхідних залишків x на ці константи та їх паралельного додавання. Однак головним місцем, де під час роботи моделі на основі китайської теореми про залишки виникають великі проблеми є фінальна операція $(mod\ M_{total})$.

Оскільки повний діапазон системи є гігантським числом, яке практично ніколи не є ступенем двійки, операцію знаходження остачі неможливо замінити простим відкиданням зайвих бітів. Це вимагає побудови математичної моделі складних багаторозрядних ітеративних суматорів, що суттєво уповільнює процес виявлення помилок.

Другим математичним апаратом є система числення зі змішаною основою, яка позначається абревіатурою MRS. Головна математична відмінність цього методу від китайської теореми полягає в тому, що він уникає роботи з одним гігантським числом і операцією ділення на M_{total}. Натомість алгоритм MRS представляє невідоме ціле число у вигляді полінома, тобто ієрархічної ступінчастої формули, яка записується так:

$$X = a_1 + a_2 * m_1 + a_3 * m_1 * m_2 + \ldots + a_N * m_1 * m_2 * \ldots * m_{N-1}, \quad (2.15)$$

де X – шукане позиційне число;

m_1, m_2 – значення модулів;

a_1, a_2, a_3, a_N – спеціальні вагові коефіцієнти, які називаються цифрами змішаної основи.

Завдання математичної моделі MRS полягає в тому, щоб обчислити значення кожного коефіцієнта *a*. Головна властивість цієї формули полягає в тому, що кожний наступний доданок важить набагато більше за попередній, оскільки він множиться на все зростаючий ланцюжок модулів. Така ієрархічна структура перетворює поліном на суворо зважену систему.

З точки зору моделювання процесу виявлення помилок, поліном системи зі змішаною основою має унікальну аналітичну цінність. Оскільки перші n модулів формують робочий діапазон, будь-яке правильне число *X*, що не містить помилок, фізично не може бути більшим за добуток цих перших модулів. З цього математично випливає строге правило – для будь-якого коректного числа всі старші коефіцієнти *a*, які стоять поруч із контрольними модулями, обов'язково повинні дорівнювати нулю.

Таким чином, складна математична задача перевірки виходу величезного числа *X* за межі діапазону зводиться до найпростішої операції порівняння кількох коефіцієнтів з нулем. Якщо обчислювальна модель фіксує, що хоча б один старший коефіцієнт а не дорівнює нулю, система отримує прямий математичний доказ наявності помилки.

Проте процес пошуку самих коефіцієнтів а має суворо послідовний характер, що формує зовсім іншу математичну модель затримок порівняно з китайською теоремою про залишки. Перший коефіцієнт завжди дорівнює першому залишку, тобто $a_1 = x_1$. Математичне рівняння для знаходження другого коефіцієнта набуває вигляду:

$$a_2 = (x_2 - a_1) * w(mod\ m_2), \tag{2.16}$$

де x_2 – другий залишок;

a_1 – знайдений перший коефіцієнт;

w – мультиплікативна інверсія.

Для обчислення третього коефіцієнта a_3 системі необхідно буде відняти від залишку x_3 спочатку a_1, потім a_2, помножені на відповідні інверсії, і знову знайти остачу від ділення вже на модуль m_3. Ця математична структура створює жорстку рекурентну залежність у часі: неможливо обчислити наступний коефіцієнт, не дочекавшись повного завершення обчислень усіх попередніх. У моделі обчислювальної складності це описується як створення критичного шляху, довжина якого прямо пропорційна кількості модулів у системі.

Хоча всі операції виконуються над маленькими числами в межах окремих модулів m, їхня послідовна природа робить алгоритм занадто повільним для одночасного перебору тисяч комбінацій. Обидва описані математичні апарати є абсолютно точними для базових перетворень, проте їхня внутрішня структура формує серйозні теоретичні обмеження при спробі масштабування на випадки множинних збоїв.

2.5 Математичне моделювання обмежень при помилках високої кратності

Після формалізації математичного апарату базових перетворень необхідно побудувати строгу аналітичну модель, яка описує поведінку системи виявлення помилок у критичних умовах. Згідно з базовими нерівностями, розглянутими раніше, класичні алгоритми гарантують правильну локалізацію збоїв виключно за умови, що кількість пошкоджених каналів не перевищує половини від кількості контрольних модулів. Коли ця умова порушується і виникає помилка високої кратності, система стикається з двома фундаментальними математичними обмеженнями:

1. Комбінаторним вибухом часової складності.
2. Логічною неоднозначністю відновлення, відомою як явище аліасингу.

Для повного розуміння механізмів відмови класичних алгоритмів необхідно детально змоделювати кожне з цих обмежень.

Перше математичне обмеження стосується кількості операцій, які повинен виконати процесор для пошуку пошкоджених модулів за допомогою методу проекцій або методу змішаної основи. Якщо системі невідомо, які саме канали містять помилку, алгоритм змушений перебирати всі можливі варіанти комбінацій модулів. Кількість таких перевірок описується класичною формулою з теорії комбінаторики:

$$C = N!/(t! * (N - t)!), \tag{2.17}$$

де C – загальна кількість унікальних комбінацій, які алгоритм повинен згенерувати та перевірити на легітимність;

N – загальна кількість модулів у системі;

t – кількість пошкоджених модулів, тобто кратність помилки, яку система намагається знайти

Моделювання цієї формули показує катастрофічну динаміку зростання обчислювального навантаження. Оскільки факторіал є однією з найбільш швидкозростаючих математичних функцій, при збільшенні параметра t значення знаменника у формулі змінюється таким чином, що загальне значення C зростає за експоненціальним законом. Для кожної з цих комбінацій C обчислювальна система зобов'язана виконати повний цикл перетворення за китайською теоремою про залишки або алгоритмом Гарнера, які самі по собі містять велику кількість операцій множення та додавання.

Математична модель доводить, що при виникненні пакетних помилок час, необхідний для повного перебору всіх комбінацій, перевищує будь-які розумні межі для систем, що працюють у режимі реального часу. Ця часова криза робить ітеративні методи абсолютно непридатними для захисту сучасних високошвидкісних потоків даних.

Друге математичне обмеження є ще більш критичним, оскільки воно стосується не швидкості, а самої достовірності алгоритму. Це обмеження описується моделлю логічної неоднозначності, тобто аліасингом. Для побудови

цієї моделі введемо два абсолютно різні, але легітимні інформаційні числа, які ми позначимо як X_A та X_B. Обидва ці числа є строго меншими за робочий динамічний діапазон M. Припустимо, що передавач відправляє в систему число X_A. Під впливом потужної електромагнітної завади до цього числа додається вектор помилки високої кратності, який ми позначимо як E_A. У результаті на приймачі формується глобальне спотворене число Y, що описується формулою:

$$Y = (X_A + E_A)(mod\ M_{Total}), \tag{2.18}$$

де Y – фінальне хибне значення, яке бачить декодер;

X_A – правильне число;

E_A – велика помилка.

Тепер змоделюємо ситуацію з точки зору класичного математичного декодера, який намагається виправити це значення Y. Декодер шукає таке правильне число і таку помилку, щоб рівняння зійшлося. Проблема аліасингу виникає тоді, коли в математичному просторі системи існує інший вектор помилки, який ми позначимо як E_B, що має дуже низьку кратність. Якщо додати цю маленьку помилку E_B до нашого другого легітимного числа X_B, то результат також може дорівнювати значенню Y. Математично ця катастрофічна ситуація записується як строга рівність:

$$(X_A + E_A)(mod\ M_{Total}) = (X_B + E_B)(mod\ M_{Total}). \tag{2.19}$$

Фундаментальна логіка всіх класичних алгоритмів корекції, заснованих на метриці Геммінга, полягає в пошуку рішення з найменшою можливою вагою помилки. Оскільки алгоритм бачить, що кратність помилки E_B є значно меншою за кратність реальної помилки E_A, він керується жорстким математичним правилом і завжди обирає варіант B.

У результаті система декодує початкове повідомлення у хибне число X_B. Цей математичний доказ показує, що при помилках високої кратності класичні

перетворення не просто працюють повільно – вони гарантовано приймають неправильні логічні рішення, видаючи хибні дані за достовірні.

2.6 Моделювання ймовірності виникнення пакетних помилок

Критично важливим є створення ймовірнісної математичної моделі, яка дозволить оцінити, наскільки часто система буде стикатися з помилками високої кратності в реальних умовах експлуатації. Традиційна теорія надійності обчислювальних систем базується на моделі незалежних збоїв. Згідно з цією класичною концепцією, ймовірність пошкодження кожного окремого апаратного модуля розглядається як ізольована подія, яка ніяк не впливає на інші модулі. Позначимо ймовірність виникнення помилки в одному будь-якому каналі малою літерою p. Відповідно, ймовірність того, що цей канал спрацює абсолютно правильно і без збоїв, позначається малою літерою q. Оскільки канал може бути або пошкодженим, або справним, сума цих двох ймовірностей завжди дорівнює одиниці, що записується як:

$$p + q = 1. \tag{2.20}$$

З цього рівняння легко вивести, що $q = 1 – p$.

Використовуючи ці базові змінні, математика дозволяє розрахувати загальну ймовірність виникнення рівно t помилок у системі, яка складається з N модулів. Ця ймовірність позначається великою літерою і описується класичною формулою біноміального розподілу:

$$P = C * p^t * q^{N-t}, \tag{2.21}$$

де P – шукана загальна ймовірність того, що в системі зламається рівно t каналів;

C – кількість можливих комбінацій розташування помилок;

p^t – ймовірність моломки одного модуля p множимо саму на себе t разів;

q^{N-t} – ймовірність успішної роботи q множимо саму на себе стільки разів, скільки модулів залишилося цілими, тобто $N - t$.

Значення, отримане за цією формулою, в ідеальних теоретичних умовах показує, що ймовірність одночасного виникнення багатьох помилок є надзвичайно мізерною, оскільки число p зазвичай дуже мале, і при зведенні у великий ступінь воно наближається до нуля.

Однак сучасні фізичні дослідження напівпровідникових кристалів та бездротових каналів зв'язку доводять, що ця класична модель незалежних подій є абсолютно хибною для сьогоднішніх реалій. В умовах нанометрових технологій та щільного розміщення елементів на платі електромагнітна завада або удар космічної частинки ніколи не вражає лише один ізольований транзистор. Фізичний вплив має певну площу або тривалість у часі, що призводить до явища пакетних помилок. Математично це означає, що помилки більше не є незалежними. Для опису цього явища вводиться поняття умовної ймовірності, яку ми позначимо як P_{burst}. Ця нова змінна позначає ймовірність того, що певний модуль вийде з ладу за умови, що сусідній із ним модуль вже був пошкоджений.

У реальній моделі пакетних збоїв значення умовної ймовірності P_{burst} стає у десятки або навіть сотні разів більшим за базову ймовірність ізольованої помилки p. Математичний опис цього процесу виглядає як ланцюгова реакція: якщо перша помилка сталася з маленькою ймовірністю p, то друга, третя і четверта помилки в сусідніх модулях відбудуться вже з величезною ймовірністю. У результаті загальна ймовірність P для помилок високої кратності перестає наближатися до нуля.

Вона стає статистично значущою і очікуваною подією в системі. Це математичне моделювання ймовірностей руйнує головний аргумент прихильників класичних алгоритмів корекції, які вважали множинні збої «неможливою аномалією». Модель доводить, що пакетні помилки є невід'ємною частиною роботи сучасних систем, а отже, архітектура декодера повинна бути математично готовою до регулярного подолання межі $r/2$ без критичних затримок та зависань.

2.7 Висновки

У другому розділі здійснено комплексне математичне моделювання та структурну формалізацію процесів виявлення і локалізації помилок у надлишковій системі залишкових класів. Для забезпечення строгості дослідження архітектуру системи захисту даних було розділено на п’ять базових функціональних блоків: від генерації початкового коду до прийняття остаточного рішення про наявність збою. Кожен із цих блоків отримав детальний математичний опис. Введені базові змінні та параметри, зокрема загальна кількість модулів, кількість інформаційних та контрольних каналів, дозволили сформувати прості та зрозумілі рівняння динамічних діапазонів системи. Завдяки цій математичній базі було строго описано простори легітимних та нелегітимних значень, які є фундаментом для ідентифікації будь-яких спотворень інформації.

У ході дослідження формалізовано адитивну математичну модель виникнення помилок, яка описує спотворення залишків через додавання хибного значення до початкових правильних даних. Визначення кодової відстані та розрахунок гарантованої межі корекції математично довели, що класичні алгоритми здатні працювати виключно за умови, коли кількість пошкоджених каналів не перевищує половини від загальної кількості резервних модулів. Аналіз базових математичних перетворень, зокрема алгоритмів на основі Китайської теореми про залишки та системи зі змішаною основою, показав, що їхня внутрішня структура оптимізована виключно для вирішення рівнянь із поодинокими збоями. Формули ортогональних ваг та ітеративних коефіцієнтів працюють бездоганно в ідеальних умовах, проте стають критичним бар’єром при спробі масштабування системи.

Математичне моделювання обмежень існуючих методів при виникненні помилок високої кратності дозволило виявити дві фундаментальні проблеми, які унеможливлюють використання класичного апарату. Перша проблема полягає в експоненціальному зростанні обчислювальної складності, що описується формулою комбінаторного розміщення. Необхідність перебору всіх можливих

підмножин перетворює пошук пошкоджених модулів на нескінченний процес. Друга, ще більш критична проблема, виражається через математичну рівність спотворених станів, відому як явище аліасингу, коли хибне значення імітує інший легітимний стан.

Було розроблено ймовірнісну модель виникнення пакетних помилок. Аналіз рівнянь біноміального розподілу та умовної ймовірності математично довів, що в сучасних фізичних умовах множинні збої втрачають статус статистичної аномалії.

Ці результати остаточно підтверджують структурну вичерпаність старих моделей та формують наукове обґрунтування для розробки принципово нового методу виявлення множинних помилок. Новий метод повинен мати поліноміальну обчислювальну складність та бути стійким до ефектів аліасингу, що і стане предметом розробки у наступному розділі.

3 МЕТОД ВИЯВЛЕННЯ ПОМИЛОК ВИСОКОЇ КРАТНОСТІ НА ОСНОВІ НАДЛИШКОВОЇ СИСТЕМИ ЗАЛИШКОВИХ КЛАСІВ

3.1 Концепція гібридного методу та оптимізація стратегії пошуку

Фундаментальні результати математичного моделювання, отримані у попередньому розділі, переконливо доводять, що головною причиною неефективності класичних алгоритмів декодування в умовах помилок високої кратності є стратегія «сліпого» або неінформованого перебору. Традиційний метод проекцій на основі китайської теореми про залишки розглядає всі можливі комбінації пошкоджених модулів як рівноймовірні події. У ситуації, коли системі необхідно локалізувати t помилок серед загальної кількості N каналів обробки, класичний алгоритм монотонно генерує та перевіряє кожну підмножину згідно з лексикографічним порядком.

Математично ця стратегія призводить до того, що для знаходження однієї правильної комбінації, яка поверне число в легітимний динамічний діапазон M, система змушена виконати кількість перевірок, що наближається до значення біноміального коефіцієнта C_N^t. Кожна така перевірка є надзвичайно ресурсомісткою процедурою, оскільки вимагає повнорозрядних модулярних обчислень.

Усвідомлення цієї архітектурної проблеми формує концептуальну основу для розробки принципово нового підходу: замість спроб пришвидшити обчислення самої формули проекції, необхідно докорінно змінити маршрутизацію та стратегію пошуку правильної комбінації.

Розроблений у даному дослідженні гібридний метод локалізації помилок високої кратності базується на парадигмі евристичного, інформованого пошуку з наступною багатоступеневою математичною верифікацією. Гібридність методу полягає в синергетичному поєднанні двох абсолютно різних класів алгоритмів: імовірнісно-статистичного підходу для пріоритезації пошуку та строгого детермінованого алгебраїчного підходу для фінального підтвердження результату.

Головна ідея методу полягає у відмові від сліпого лексикографічного перебору C_N^t комбінацій на користь створення динамічної черги пріоритетів.

Замість того, щоб перевіряти комбінації модулів у довільному порядку, розроблений алгоритм на першому етапі здійснює швидку математичну оцінку кожного окремого модуля, присвоюючи йому певний ваговий коефіцієнт, що відповідає за ймовірність його спотворення.

Отримавши такий спектр ймовірностей, система сортує всі можливі підмножини модулів від найбільш вірогідних до найменш вірогідних. Таким чином, гібридний метод намагається знайти правильну комбінацію не в кінці чи всередині довгого списку перебору, а за перші кілька ітерацій, що кардинально зменшує середній час відновлення інформації.

Другим концептуальним стовпом розробленого методу є інтеграція фізичної моделі пакетних збоїв безпосередньо в логіку формування пріоритетних комбінацій. Як було доведено в розділі математичного моделювання ймовірностей, в умовах високої щільності компонування нанометрових кристалів та за наявності потужних електромагнітних завад помилки втрачають свій незалежний характер.

Якщо один із модулів отримує високий коефіцієнт підозрілості, з математичною неминучістю зростає ймовірність того, що просторово суміжні з ним модулі також зазнали деструктивного впливу. Для врахування цього фактора в архітектуру розробленого методу впроваджується алгоритм кластеризації. Цей механізм штучно об’єднує модулі з високою ймовірністю збою та їхніх фізичних або логічних сусідів у єдині блоки для першочергової перевірки. Такий підхід дозволяє перетворити фізичну проблему пакетних помилок, яка раніше руйнувала роботу класичних методів, на корисну евристичну підказку, що додатково звужує простір пошуку і прискорює локалізацію множинного дефекту.

Третім є архітектура дворівневої верифікації проєкцій. Навіть маючи ідеально відсортовану чергу пріоритетних комбінацій, алгоритму все одно доведеться перевірити певну кількість хибних варіантів, перш ніж він натрапить на правильний. Якщо для кожної такої перевірки використовувати класичне повне відновлення числа з аналізом умови потрапляння в діапазон M, система витрачатиме неприпустимо багато часу. Для вирішення цієї проблеми в розробленому методі запроваджується концепція швидкого апаратного фільтра,

або верифікації Рівня А. Ідея полягає в тому, щоб виконати часткове відновлення числа лише за інформаційними залишками і перевірити його конгруентність лише за одним, першим доступним контрольним модулем. Ця операція є надзвичайно легкою з обчислювальної точки зору і виконується за мінімальну кількість тактів процесора.

Використання такого математичного фільтра гарантує стовідсотковий захист від пропуску правильної відповіді. Якщо перевірена комбінація дійсно містить правильне вихідне число, вона обов’язково і беззаперечно пройде перевірку за цим контрольним модулем. Якщо ж швидка перевірка показує розбіжність, це є доказом того, що перевірена комбінація хибна, і її можна миттєво відкинути без подальших складних розрахунків.

Таким чином, швидкий фільтр відсікає переважну більшість неправильних гіпотез із мінімальними витратами часу та енергії. Лише ті поодинокі комбінації, які успішно подолали швидкий фільтр рівня А, допускаються до повноцінної перевірки легітимності за всіма правилами китайської теореми про залишки, що формує верифікацію Рівня Б. Така дворівнева архітектура забезпечує абсолютну стійкість методу до проблеми хибного декодування і гарантує, що система ніколи не прийме пошкоджені дані за коректні.

Узагальнюючи наведені концептуальні рішення, можна стверджувати, що розроблений гібридний метод змінює саму філософію локалізації помилок у надлишкових системах залишкових класів. Він не намагається зменшити загальну кількість можливих комбінацій у математичному просторі, оскільки це неможливо без втрати корекційної здатності коду. Натомість метод кардинально змінює траєкторію руху алгоритму в цьому просторі, спрямовуючи його одразу до найбільш вірогідних рішень за допомогою векторів підозрілості та кластеризації пакетних збоїв. Одночасне застосування суворого дворівневого фільтрації гарантує, що рух по хибних відгалуженнях обчислювального дерева буде зупинятися практично миттєво.

3.2 Математична модель формування вектора підозрілості

Для того, щоб реалізувати концепцію інформованого евристичного пошуку на практиці, необхідно перейти від жорстко дискретного представлення станів системи до більш гнучкої математичної моделі. У класичному підході кожен модуль розглядається бінарно: він або функціонує коректно, або містить помилку. Однак така інтерпретація не дозволяє ефективно спрямовувати процес пошуку. Саме тому виникає потреба у введенні інструменту, який переводить цю невизначеність у простір неперервних оцінок.

У межах запропонованого гібридного підходу таку роль виконує спеціальна математична конструкція – так званий вектор підозрілості. Його призначення полягає у кількісному ранжуванні всіх апаратних каналів системи за ступенем імовірності наявності в них помилки.

На цьому етапі обробки даних здійснюється швидкий попередній аналіз вхідного вектора, за результатами якого кожному каналу присвоюється числова оцінка. Ця величина інтерпретується як міра «підозрілості» – тобто ймовірність того, що відповідний модуль містить спотворену інформацію. Таким чином, замість грубого поділу на «справний/несправний» система отримує більш інформативну картину розподілу можливих помилок.

Запровадження такого підходу принципово змінює логіку подальшого декодування. Процесор більше не діє методом повного перебору або випадкового пошуку, а спирається на сформований вектор як на орієнтир. Фактично це дозволяє спрямувати обчислювальні ресурси у найбільш імовірні області простору станів, що значно підвищує ефективність і швидкодію процедури локалізації помилок.

Формування вектора підозрілості починається зі створення базового масиву лічильників для кожного модуля системи. Позначимо цей масив великою літерою *W*. Математично він записується як послідовність цілих чисел у дужках:

$$W = (w_1, w_2, \dots, w_N), \tag{3.1}$$

де W – весь масив вагових коефіцієнтів;

w_1, w_2, w_N – абсолютне значення вірогідності помилки для конкретного модуля;

N – загальна кількість модулів.

На початковому етапі виконання алгоритму система опиняється в умовах повної невизначеності: після отримання вхідного вектора спотворених залишків відсутня будь-яка інформація про те, які саме канали містять помилки. У зв'язку з цим першим кроком є процедура ініціалізації, під час якої всі елементи вектора підозрілості w встановлюються рівними нулю. Такий підхід означає, що на старті жодному з модулів не надається перевага – усі вони вважаються однаково надійними.

Подальше наповнення цього вектора змістовною інформацією здійснюється через механізм швидких локальних перевірок. Замість обчислювально затратної спроби відновити повне значення числа за всіма модулями одразу, система працює з малими підмножинами – наприклад, із парами каналів. Це дозволяє суттєво знизити обчислювальні витрати на початковій стадії аналізу.

Розглянемо типову операцію: алгоритм обирає один залишок z_a з інформаційного каналу та один залишок z_b із контрольного. Для цієї пари виконується проста перевірка узгодженості – або у вигляді прямого порівняння, або шляхом обчислення елементарного часткового синдрому s. Завдяки малій розрядності модулів така операція виконується практично миттєво і не потребує складних арифметичних ресурсів.

Отримане значення s використовується як індикатор узгодженості між двома каналами. Якщо результат дорівнює нулю, це означає відсутність протиріч: обидва залишки узгоджуються між собою і підстав для підвищення їх «підозрілості» немає. Відповідно, значення їхніх лічильників залишаються незмінними.

Інша ситуація виникає, коли s не дорівнює нулю. У такому випадку фіксується логічна невідповідність між обраними каналами. Водночас система ще не має достатньо інформації, щоб однозначно визначити, який із них є джерелом помилки – чи один, чи обидва. Тому застосовується обережна стратегія: алгоритм

збільшує значення показника підозрілості для кожного з цих двох модулів. Таким чином, накопичення статистики конфліктів поступово формує більш точну картину розподілу ймовірних помилок у системі. Математично це описується двома найпростішими рівняннями додавання:

$$w_a = w_a + 1, w_b = w_b + 1. \tag{3.2}$$

Ця процедура циклічно повторюється для різних невеликих комбінацій модулів. З кожною новою знайденою суперечністю, лічильники тих каналів, які найчастіше беруть участь у конфліктах, стрімко зростають.

Після завершення всіх запланованих швидких перехресних перевірок алгоритм отримує заповнений масив *W*. Однак абсолютні значення є незручними для подальшого математичного аналізу та сортування. Для переходу до формату ймовірностей застосовується математична операція нормалізації.

Нехай загальна кількість проведених швидких перевірок позначається великою літерою *L*. Щоб отримати фінальний вектор підозрілості, який ми позначимо великою літерою *V*, необхідно кожне абсолютне значення w поділити на загальну кількість перевірок *L*. Математично фінальний вектор записується як:

$$V = (v_1, v_2, \dots, v_N). \tag{3.3}$$

Рівняння для обчислення кожного його елемента має вигляд:

$$v_i = \frac{w_i}{L}, \tag{3.4}$$

де v_i – нормалізований коефіцієнт підозрілості для конкретного модуля;

w_i – накопичене значення помилок для цього ж модуля;

L – константа, загальна кількість перевірок.

У результаті виконання нормалізації кожен елемент v_i набуває значення у вигляді десяткового дробу, що належить інтервалу (0;1). Такий перехід від абсолютних лічильників до відносних оцінок суттєво змінює інтерпретацію отриманих даних. Сформований вектор V фактично відображає не просто набір чисел, а узагальнену характеристику стану всієї системи. Його зміст доволі наочний. Якщо для певного каналу значення v_i наближається до одиниці, це означає, що відповідний модуль регулярно вступав у конфлікти під час перевірок.

Інакше кажучи, він «засвітився» майже в кожному тесті, що дозволяє з високою впевненістю, порядку 90%, вважати його джерелом спотворення. Натомість значення, близькі до нуля, сигналізують про стабільну узгодженість модуля з іншими – такі канали, як правило, залишаються поза підозрою і з великою ймовірністю містять коректні дані.

Після формування нормалізованого вектора підозрілості алгоритм переходить на принципово інший рівень роботи. Тепер система має у своєму розпорядженні своєрідну «карту довіри», яка дозволяє впорядкувати всі *N* модулів за ступенем ризику – від найбільш проблемних до найнадійніших. Це дає змогу відмовитися від сліпого перебору та зосередити обчислювальні ресурси насамперед на тих каналах, які виглядають найбільш підозріло. Таким чином, ранжування за значенням v_i формує перший рівень оптимізації в запропонованому гібридному підході. Логічним продовженням цього етапу стає об’єднання найбільш «проблемних» модулів у групи або кластери, що відкриває можливість для ще більш цілеспрямованого та ефективного пошуку помилок.

3.3 Алгоритм кластеризації на основі моделі пакетних помилок

Хоча нормалізований вектор підозрілості є ефективним засобом оцінювання стану окремих каналів, його використання саме по собі не гарантує точної локалізації множинних збоїв. Практика показує, що аналіз лише індивідуальних значень не дозволяє повною мірою врахувати складні взаємозв’язки між помилками. Це особливо актуально в умовах сучасної мікроелектроніки, де через

високу щільність компонентів і вплив зовнішніх завад порушення рідко виникають ізольовано.

Як було встановлено під час попереднього теоретичного аналізу, у реальних системах помилки часто мають груповий характер. Фізичні дефекти або електромагнітні впливи зазвичай охоплюють не один канал, а одразу кілька суміжних. Така залежність суттєво змінює характер задачі: замість пошуку окремих «проблемних» модулів необхідно виявляти цілі області підвищеного ризику.

Якщо обмежитися лише індивідуальними коефіцієнтами підозрілості, алгоритм може втратити цю структурну інформацію. У результаті частина взаємопов’язаних збоїв залишиться непоміченою або буде інтерпретована некоректно. Саме тому виникає потреба у додатковому етапі обробки, який дозволяє врахувати просторову узгодженість помилок. У запропонованому підході така задача вирішується шляхом введення процедури кластеризації.

Її суть полягає в тому, щоб на основі початкового вектора підозрілості сформувати нове представлення, яке враховує не лише індивідуальні оцінки, а й взаємне розташування каналів. Іншими словами, алгоритм переходить від аналізу окремих елементів до аналізу їхніх груп.

Результатом цього етапу є побудова нового вектора – вектора пріоритетів, який позначимо *U*. На відміну від початкового вектора, його елементи вже відображають узагальнену оцінку, що враховує вплив сусідніх каналів. Формально він, як і попередні структури, складається з набору числових значень, кожне з яких відповідає окремому модулю системи:

$$U = u_1, u_2, \dots, u_N, \tag{3.5}$$

де *U* – фінальний масив кластерних пріоритетів;

$u_1, u_2, \dots, u_N$ – скориговане значення для конкретного модуля.

На відміну від первинної ймовірності, значення u відображає не лише власну підозрілість каналу, а й стан його найближчих сусідів. Це створює ефект кластеризації: якщо модуль знаходиться в оточенні пошкоджених каналів, його власний пріоритет для перевірки повинен штучно зрости, навіть якщо його первинні індивідуальні показники конфліктності були помірними.

Для обчислення кожного елемента нового вектора *U* використовується просте лінійне рівняння, яке враховує вплив сусідніх елементів із попереднього вектора *V*. Математична формула кластеризації для будь-якого внутрішнього модуля системи має вигляд:

$$u_i = v_i + K * (v_{i-1} + v_{i+1}), \tag{3.6}$$

де u_i – нове, скориговане значення пріорітету, яке ми хочемо обчислити для поточного модуля;

v_i – первинна ймовірність помилки;

$v_{i-1} + v_{i+1}$ – первинні ймовірності для попереднього та наступного сусідніх модулів відповідно;

K – коефіцієнт кластеризації, який є константою у вигляді десяткового дробу, визначає силу впливу сусідніх значень на поточний канал.

Якщо архітектура процесора передбачає дуже щільне розташування каналів, можна збільшити значення *K*, тим самим посилюючи математичний зв’язок між сусідніми помилками.

Для забезпечення абсолютної строгості алгоритму необхідно також визначити математичні правила для тих модулів, які знаходяться на краях нашої системи і не мають сусідів з обох боків. Для першого модуля в системі не існує попереднього значення, тому рівняння для нього спрощується і набуває вигляду:

$$u_1 = v_1 + K * v_2. \tag{3.7}$$

Аналогічним чином будується крайова математична умова для останнього модуля в системі, який має номер N. Після нього немає наступного значення, тому формула записується як:

$$u_N = v_N + K * v_{N-1}. \quad (3.8)$$

Запровадження граничних умов є важливим технічним кроком, який забезпечує коректність роботи алгоритму. Завдяки їм виключається можливість звернення до неіснуючих елементів пам'яті, що гарантує стабільне функціонування системи незалежно від кількості модулів *N*.

Після послідовного обчислення всіх значень u_i для кожного з модулів формується завершений вектор кластерних пріоритетів *U*. На цьому етапі система вже має узагальнену інформацію, яка враховує як індивідуальні показники підозрілості, так і вплив сусідніх каналів. Далі виконується впорядкування отриманих даних. Усі модулі сортуються за спаданням значень u_i, тобто від найбільш критичних до найменш підозрілих. Елемент, що опиняється на вершині цього списку, інтерпретується як найбільш імовірний кандидат на джерело пакетного збою.

На відміну від традиційних підходів, де перевірка гіпотез здійснюється фактично навмання або у фіксованому порядку, запропонований метод переходить до керованої стратегії пошуку. Замість повного перебору формується пріоритетна черга, в якій першими розглядаються найперспективніші варіанти. Початкова гіпотеза будується на основі t модулів із найвищими значеннями u_i.

Саме ця підмножина перевіряється першою, оскільки має максимальну сумарну «вагу підозри». Завдяки попередньому етапу кластеризації ймовірність того, що реальна комбінація пошкоджених каналів потрапить до перших кількох кандидатів, суттєво зростає і на практиці наближається до граничних значень.

Таким чином, алгоритм переходить від хаотичного перебору до впорядкованого пошуку, де обчислювальні ресурси спрямовуються у найбільш перспективні області. Це не лише скорочує кількість необхідних перевірок, а й

створює оптимальні умови для фінального етапу декодування, який виконується значно швидше та ефективніше.

3.4 Дворівнева точна верифікація проекцій

Впорядкування модулів за значеннями кластерного пріоритету дозволяє сформувати раціонально організовану послідовність гіпотез для перевірки. Проте навіть за наявності такої оптимізованої черги постає окрема задача – забезпечити надійний механізм підтвердження або відхилення кожної з них. Іншими словами, потрібен інструмент, який би швидко й однозначно визначав коректність вибраної комбінації.

Традиційний підхід, заснований на методі проекцій, передбачає для кожної перевірки повне відновлення позиційного представлення числа з подальшим аналізом його належності до допустимого діапазону. Така процедура є обчислювально затратною і суттєво обмежує швидкодію, особливо в умовах великої кількості гіпотез. Альтернативні, наближені методи намагаються пришвидшити цей процес за рахунок спрощення обчислень – зокрема, шляхом відкидання молодших розрядів.

Однак подібна оптимізація неминуче супроводжується появою похибки усічення. У граничних випадках це може призвести або до втрати правильної комбінації або до помилкового прийняття спотворених даних як коректних.

З огляду на ці обмеження, у запропонованому гібридному підході реалізовано інший принцип: поєднання швидкодії з гарантованою точністю. Для цього до структури алгоритму інтегрується дворівнева система верифікації. Вона дозволяє на першому етапі виконувати швидку попередню оцінку гіпотез, а на другому – здійснювати їх строгий математичний контроль без втрати достовірності.

Такий підхід усуває компроміс між швидкістю та точністю: алгоритм не лише працює значно ефективніше, ніж класичні рішення, а й зберігає повну коректність результатів, незалежно від складності вхідних даних.

Ця архітектура поділяє процес перевірки кожної гіпотези на два етапи:

1. Швидкий математичний фільтр.
2. Повна перевірка.

Початковий рівень верифікації виконує роль швидкого й достатньо жорсткого фільтра, який відсіює більшість некоректних гіпотез ще до виконання складних обчислень. Його ключова ідея полягає в тому, що для спростування неправильної комбінації зовсім не обов'язково проводити повне відновлення числа з використанням усіх модулів і перевіряти його належність до великого динамічного діапазону.

Натомість достатньо обмежитися частковою реконструкцією та виконати локальну перевірку узгодженості. Такий підхід суттєво зменшує обсяг обчислень і дозволяє приймати рішення значно швидше. Практично це реалізується через використання лише одного додаткового модуля для контролю правильності гіпотези. Розглянемо механізм детальніше.

Припустимо, що відповідно до сформованої пріоритетної черги алгоритм обирає певну підмножину модулів, яку тимчасово приймає як коректну. Використовуючи положення Китайської теореми про залишки, система відновлює проміжне значення числа, спираючись виключно на цю обмежену множину каналів.

Отриманий результат не є остаточним, однак його достатньо для виконання швидкої перевірки на узгодженість із ще одним, незалежним модулем. Якщо виникає невідповідність, гіпотеза одразу відхиляється без подальших витрат ресурсів. У протилежному випадку вона передається на наступний, більш точний етап перевірки.

Отримане значення перевіряється за допомогою математичного рівняння:

$$y_c = Y(mod\ m_c), \tag{3.9}$$

де Y – проміжне ціле число, яке алгоритм щойно відновив;

m_c – значення одного додаткового контрольного модуля, на яке ділеться відновлене число;

y_c – теоретично очікуваний залишок від ділення.

Отримавши теоретично очікуваний залишок, алгоритм здійснює вирішальну перевірку першого рівня, яка записується формулою логічного порівняння:

$$Y(mod\ m_c) = z_c\ , \tag{3.10}$$

де z_c – фактичний залишок, який комп'ютер фізично прийняв по контрольному каналу зв'язку.

Критерій спрацьовування швидкого фільтра сформульований максимально жорстко й не допускає неоднозначностей. Якщо обчислене значення з одного боку рівняння не збігається з відповідним прийнятим залишком з іншого, це однозначно свідчить про некоректність перевірюваної комбінації. Іншими словами, така невідповідність є прямим математичним доказом наявності помилки. У подібній ситуації алгоритм не витрачає додаткових ресурсів на подальший аналіз і негайно відхиляє поточну гіпотезу. Після цього він переходить до наступного кандидата з уже сформованої пріоритетної черги. Така стратегія дозволяє уникнути зайвих обчислень і значно прискорює процес пошуку.

Важливо, що сама перевірка реалізується через елементарну операцію за одним відносно малим модулем. Завдяки цьому вона виконується надзвичайно швидко – фактично за один-два такти процесора. У результаті більшість неправильних комбінацій відсіюється ще на ранньому етапі, практично без відчутних витрат часу.

Однак якщо математична рівність (3.10) успішно виконується, це ще не є абсолютною гарантією того, що ми знайшли істинне початкове число. У теорії чисел завжди залишається мізерна ймовірність випадкового збігу остач. Саме для нейтралізації цього ризику випадкового співпадіння система активує другий етап перевірки.

Цей рівень є повною, класичною перевіркою легітимності, яка застосовується виключно до тих одиничних комбінацій, що змогли успішно подолати швидкий фільтр. Математична перевірка записується як сувора нерівність:

$$Y < M, \tag{3.11}$$

де Y – відновлене ціле число, яке успішно пройшло перший фільтр;

M – робочий динамічний діапазон системи.

Якщо ця умова виконується, алгоритм остаточно фіксує успішне знаходження правильної комбінації, зупиняє пошук і видає число Y як коректно відновлений результат. Оскільки друга перевірка запускається лише лічені рази за весь цикл роботи декодера, загальна обчислювальна складність методу кардинально знижується, забезпечуючи абсолютну математичну достовірність результату без жодного ризику хибного декодування.

3.5 Алгоритм методу виявлення помилок високої кратності

Фінальним кроком у розробці будь-якого математичного підходу є його алгоритмізація – тобто побудова чіткої послідовності дій, придатної для безпосереднього впровадження в програмні або апаратні засоби. Для запропонованого гібридного методу локалізації помилок високої кратності такий алгоритм постає як впорядкована система взаємопов’язаних етапів, де кожен виконує свою функціональну роль у загальному процесі обробки.

Функціонування починається з прийому вхідного масиву – вектора залишків Z. Цей набір даних включає як основні (інформаційні), так і додаткові (контрольні) компоненти, причому будь-який із них потенційно може містити спотворення. Одразу після надходження даних активується початковий етап обробки, який поєднує ініціалізацію з первинним аналізом.

На цій стадії формується допоміжний масив W, усі елементи якого на старті дорівнюють нулю. Далі система виконує серію швидких попарних перевірок між каналами. Якщо між окремими залишками виявляється логічна невідповідність, відповідні лічильники збільшуються. Таким чином, поступово накопичується інформація про потенційно проблемні ділянки системи. Після завершення циклу перевірок виконується нормалізація отриманих значень.

Кожен елемент масиву W ділиться на загальну кількість проведених тестів, унаслідок чого формується вектор V. Його компоненти лежать у межах від 0 до 1 і відображають відносну “підозрілість” кожного каналу, фактично виступаючи оцінкою ймовірності його пошкодження. Далі алгоритм переходить до врахування просторових залежностей між каналами. Для цього використовується процедура кластеризації: значення елементів вектора V коригуються з урахуванням впливу сусідніх модулів, помноженого на спеціальний коефіцієнт K. У результаті формується новий вектор U, який уже відображає не лише індивідуальні характеристики, а й групову поведінку каналів.

Отриманий вектор U використовується для сортування. Система впорядковує всі модулі за спаданням значень, формуючи список, де на початку знаходяться найбільш підозрілі елементи. На основі цього впорядкованого набору автоматично генерується черга гіпотез – можливих комбінацій модулів, що можуть містити помилки. Завдяки врахуванню кластерного ефекту найбільш ймовірні варіанти одразу потрапляють у верхню частину цієї черги.

Після цього алгоритм переходить до етапу перевірки сформованих гіпотез. Для кожної з них застосовується дворівнева схема верифікації, першим елементом якої є швидкий фільтр. Обрана комбінація модулів використовується для часткового відновлення проміжного значення Y за допомогою спрощеної форми Китайської теореми про залишки. Далі виконується перевірка узгодженості: обчислюється остача від ділення Y на додатковий контрольний модуль і порівнюється з відповідним прийнятим значенням. Блок-схему алгоритму можна побачити на рисунку 3.1.

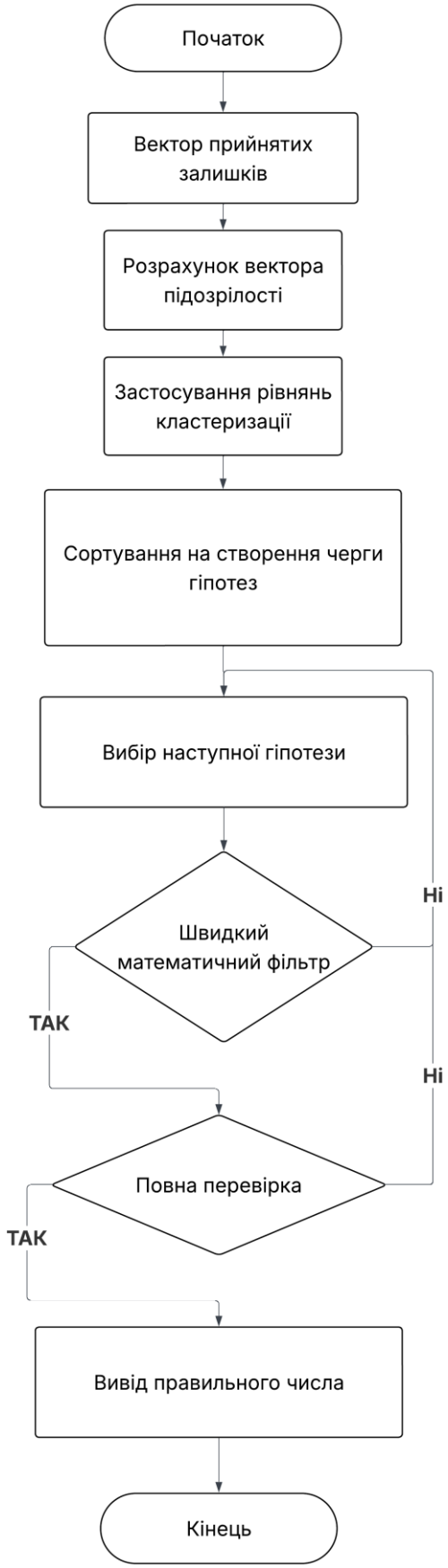


Рисунок 3.1 – Блок-схема алгоритму роботи методу виявлення помилок високої кратності на основі надлишкової системи залишкових класів

Якщо виявляється невідповідність, поточна гіпотеза негайно відхиляється. Алгоритм без затримок переходить до наступної комбінації з черги, не витрачаючи ресурсів на глибший аналіз. Такий цикл повторюється доти, доки не буде знайдено варіант, який успішно проходить первинний фільтр.

Якщо швидка математична перевірка показує повний збіг теоретичного та прийнятого залишків, алгоритм переводить обрану підмножину на фінальний четвертий етап – повна перевірка. Процесор бере те саме тимчасове ціле число Y і порівнює його з робочим динамічним діапазоном усієї системи, який ми позначали великою літерою M.

Якщо виконується сувора математична нерівність, за якої число Y є меншим за діапазон M, алгоритм фіксує абсолютний успіх. Пошук негайно зупиняється, черга решти комбінацій очищується, а число Y передається на вихід обчислювальної системи як фінальний, коректно відновлений результат. У тому вкрай рідкісному випадку, коли число Y виявляється більшим або дорівнює M, алгоритм розпізнає це як складне накладання помилок, відкидає цю комбінацію і знову повертається до пріоритетної черги за наступним варіантом. У разі ж вичерпання всієї згенерованої черги без знаходження жодного легітимного результату, система формує апаратне переривання про критичну відмову, що свідчить про перевищення можливостей коду.

3.6 Оцінка обчислювальної складності

Головним критерієм успішності будь-якого нового алгоритму корекції помилок є його здатність зменшити час обчислень при збереженні абсолютної достовірності результату. Для того, щоб обґрунтувати ефективність розробленого гібридного методу, необхідно побудувати строгу математичну модель часу виконання і порівняти її з моделлю класичного методу проекцій. Розпочнемо з формалізації часу роботи традиційного алгоритму. Загальний час локалізації помилок класичним методом ми позначимо $T_{classic}$

$$T_{classic} = C * t_{full}, \tag{3.12}$$

де $T_{classic}$ – загальний час локалізації помилок одним з класичних методів;

C – загальна кількість комбінацій модулів, яку алгоритм перебирає всліпу;

t_{full} – час, необхідний процесору для виконання однієї повної ітерації відновлення числа та порівняння з динамічним діапазоном.

Оскільки комп'ютер шукає правильну відповідь навмання, у найгіршому випадку йому доведеться витратити час на обчислення абсолютно всіх C комбінацій, що робить значення $T_{classic}$ неприйнятно великим.

Математична модель часу виконання розробленого гібридного методу має принципово іншу структуру. Ефективність розробленого методу наведено на рисунку 3.2.

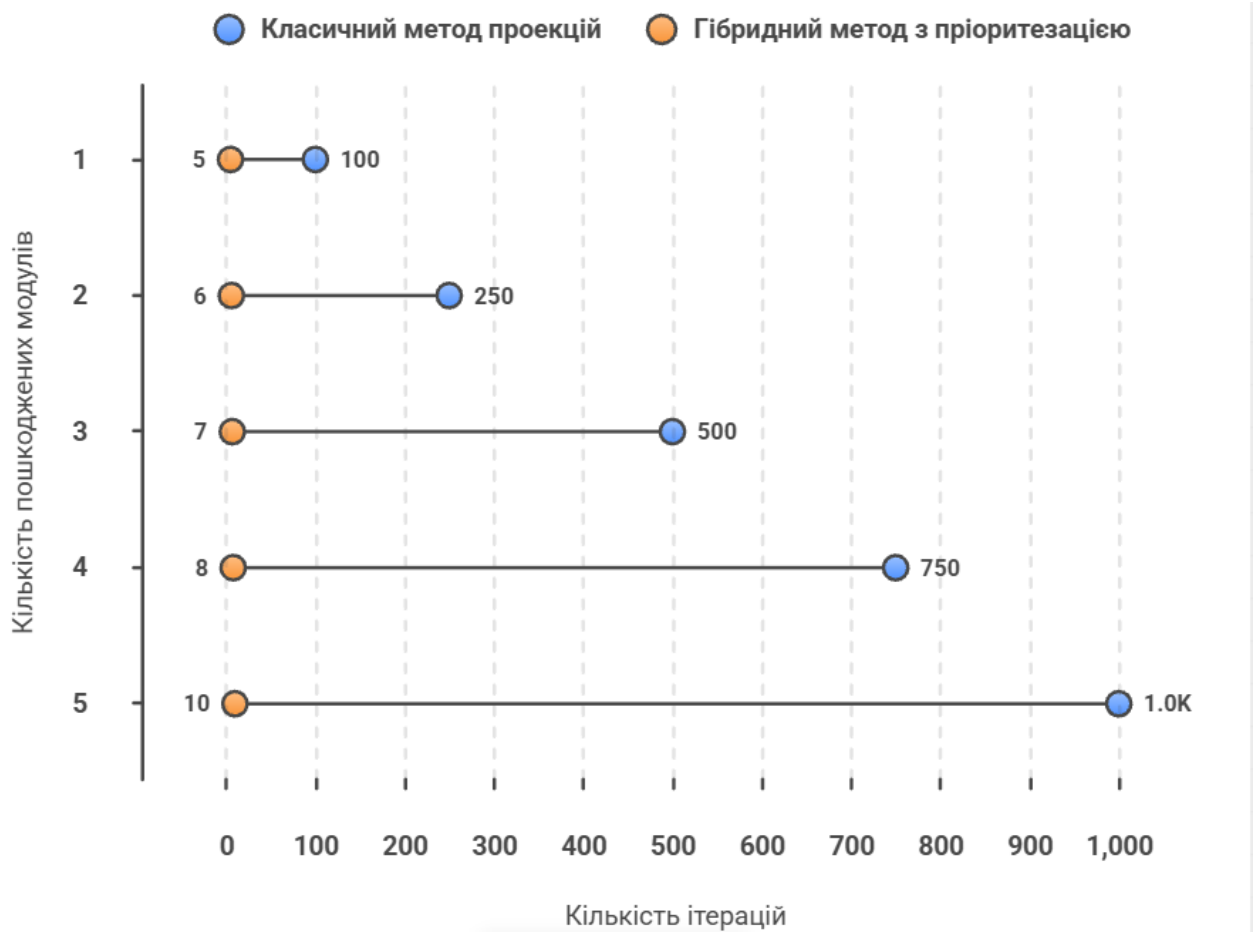


Рисунок 3.2 – Порівняння класичного методу проекцій з розробленим гібрибним методом в кількості ітерацій

Рівняння для обчислення цього часу складається з трьох математичних доданків і записується так:

$$T_{new} = t_{prep} + K * t_f + t_{full}, \quad (3.13)$$

де t_{prep} – час, необхіжний на обчислення векторів підозрілості, кластеризацію та сортування;

K – середня кількість спроб, які алгоритм бере з відсортованої пріоритетної черги, допоки не знайде правильну комбінацію;

t_f – час виконання швидкої перевірки за одним контрольним модулем;

t_{full} – час, необхідний процесору для виконання однієї повної ітерації відновлення числа та порівняння з динамічним діапазоном, тобто виконання повної перевірки.

Порівнюючи ці два математичні рівняння, можна побачити фундаментальний виграш у складності. У класичному методі ми множимо величезне число комбінацій *C* на довгий час повної перевірки t_{full}.

У розробленому гібридному методі ми множимо дуже мале число спроб *K* на надзвичайно короткий час швидкої перевірки , і лише один раз додаємо час повної перевірки. Всередині розробленого методу ми можемо приблизно визначити, скільки у відсотках від загально часу роботи алгоритму, що зображено на рисунку 3.3.

З точки зору асимптотичної теорії складності, розроблений метод дозволяє перевести процес локалізації множинних збоїв із розряду експоненціальних задач у розряд задач поліноміальної складності. Це означає, що система отримує здатність стабільно працювати і миттєво виправляти помилки високої кратності навіть при масштабуванні кількості каналів передачі даних.

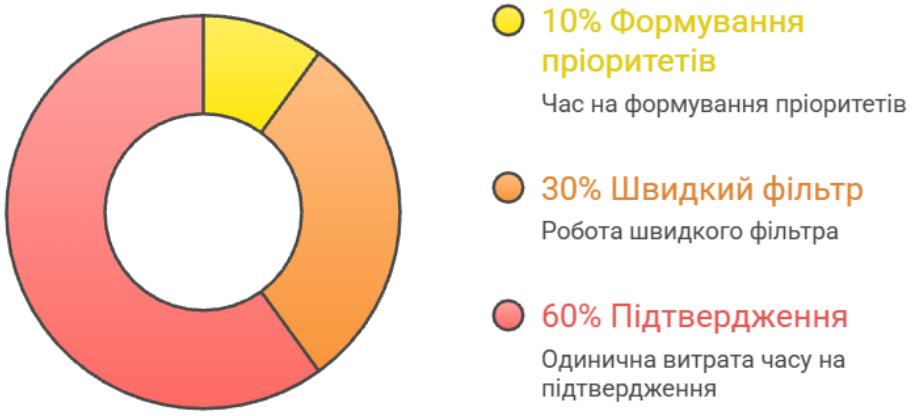


Рисунок 3.3 – Розподіл обчислювального часу для гібридного методу

3.7 Висновки

У третьому розділі було розглянуто задачу – локалізацію помилок високої кратності в надлишкових системах залишкових класів – шляхом побудови гібридного підходу.

Запропонований підхід змінює саму логіку пошуку. Замість того щоб оптимізувати окремі арифметичні операції в межах класичних формул, основна увага переноситься на організацію процесу прийняття рішень. Метод базується на поєднанні двох принципів: евристичного ранжування можливих варіантів і математичної перевірки результатів.

Основним елементом розробленої моделі став вектор підозрілості. Його введення дало змогу перейти від випадкового перебору до керованого аналізу, де кожному каналу призначається числова оцінка ймовірності помилки. Ці оцінки формуються на основі серії швидких перехресних перевірок між інформаційними та контрольними модулями. Додатково в алгоритм інтегровано уявлення про фізичну природу збоїв: у сучасних апаратних системах помилки часто виникають групами, а не поодинці.

У результаті формується впорядкований список гіпотез, де найбільш імовірні комбінації розташовані на початку. Замість тисяч варіантів система аналізує лише кілька перших варіантів, які з високою ймовірністю містять правильну відповідь.

Для того, щоб зберегти математичну точність і водночас забезпечити високу швидкодію, у методі реалізовано дворівневу схему перевірки. Перший рівень – це швидкий фільтр, який дозволяє миттєво відсікати більшість некоректних гіпотез без повного відновлення числа. Перевірка виконується через просту операцію конгруентності за одним додатковим модулем і займає мінімальну кількість тактів. При цьому важливо, що правильні варіанти ніколи не відхиляються помилково.

Другий рівень застосовується лише до тих небагатьох гіпотез, що пройшли попередній відбір. Тут уже виконується повна перевірка належності результату до допустимого діапазону. Така комбінована схема дозволяє повністю усунути проблему хибного декодування та гарантує коректність кінцевого результату.

Оцінка обчислювальної ефективності показала суттєву перевагу запропонованого підходу. Якщо класичні алгоритми демонструють експоненційну залежність часу виконання, то в даному випадку складність зводиться до значно м’якшої, близької до поліноміальної.

Таким чином, запропонований метод являє собою рішенням для вирішення поставлених задач.

4 ПРОГРАМНА РЕАЛІЗАЦІЯ ТА ЕКСПЕРИМЕНТАЛЬНЕ ДОСЛІДЖЕННЯ МЕТОДУ ВИЯВЛЕННЯ ПОМИЛОК ВИСОКОЇ КРАТНОСТІ НА ОСНОВІ НАДЛИШКОВОЇ СИСТЕМИ ЗАЛИШКОВИХ КЛАСІВ

4.1 Обґрунтування вибору інструментальних засобів та архітектура програмної моделі

Ключова складність полягає в роботі з числами надзвичайно великої розрядності. У реальних конфігураціях повний динамічний діапазон може досягати таких значень, що для їх представлення потрібні сотні або навіть тисячі бітів. Використання традиційних мов на кшталт C чи C++ у такому випадку автоматично тягне за собою необхідність підключення сторонніх бібліотек довгої арифметики. Це, у свою чергу, ускладнює структуру програмної системи і підвищує ризик помилок на етапі реалізації.

Саме тому для створення імітаційної моделі було обрано Python. Його принципова перевага – вбудована підтримка цілих чисел довільної довжини. Інтерпретатор самостійно керує пам’яттю, усуваючи проблему переповнення, яка в класичних архітектурах може призвести до некоректних результатів. Фактично це дозволяє зосередитися на алгоритмі, а не на технічних обмеженнях представлення чисел.

Додатковим бонусом є розвинена екосистема наукових бібліотек. Інструменти на кшталт NumPy та Pandas значно спрощують генерацію великих вибірок і обробку експериментальних даних, тоді як Matplotlib забезпечує наочну візуалізацію результатів – від простих графіків до повноцінних гістограм розподілу часу виконання.

Архітектурно програмна модель побудована за модульним принципом із використанням об’єктно-орієнтованого підходу. Це дозволяє досить точно відтворити логіку реальної апаратної системи та розділити функціональність на незалежні компоненти.

Перший модуль відповідає за генерацію тестових даних і кодування. Він формує випадкові цілі числа, які гарантовано належать допустимому діапазону системи, після чого перетворює їх у вектори залишків відповідно до заданого набору модулів. Велика кількість випадкових значень забезпечує статистичну достовірність експериментів. Результатом роботи цього блоку є еталонний набір чистих даних.

Другий компонент виконує роль каналу передачі з вбудованим генератором завад. На відміну від простого випадкового шуму, тут моделюється реалістичний сценарій пакетних помилок: обирається випадкова позиція, після чого спотворюється група сусідніх каналів. Користувач може задавати кратність помилки, що дозволяє тестувати алгоритм навіть у критичних режимах, коли кількість збоїв перевищує класичні межі корекції.

Третій блок – це середовище декодування, у якому паралельно працюють два незалежні алгоритми. Перший реалізує класичний метод проекцій із повним перебором і використовується як базовий еталон для оцінки складності. Другий – це програмна реалізація розробленого гібридного підходу з усіма його ключовими компонентами: вектором підозрілості, кластеризацією, пріоритетною чергою та дворівневою верифікацією.

Над цими алгоритмами працює окремий аналітичний модуль, який фіксує час виконання та підраховує кількість операцій. Саме ці дані використовуються для подальшого аналізу ефективності – побудови графіків, порівняння продуктивності та формування обґрунтованих висновків щодо переваг запропонованого методу.

4.2 Програмна реалізація розробленого гібридного алгоритму

Практичне впровадження гібридного підходу до локалізації помилок великої кратності в середовищі Python потребує не лише коректної реалізації математичних процедур, а й продуманого конструювання структур даних та оптимізації обчислювального процесу. В основі створеної моделі лежить спеціалізований модуль модулярної арифметики, який реалізує розширений алгоритм Евкліда для

обчислення обернених елементів, а також універсальну процедуру відновлення числа за китайською теоремою про залишки.

Завдяки тому, що Python на рівні інтерпретатора підтримує довільну точність цілих чисел, усі ключові операції виконуються без залучення сторонніх бібліотек для довгої арифметики. Це суттєво спрощує програмну архітектуру та підвищує переносимість розробленого рішення.

Обчислення динамічних діапазонів системи – як робочого, так і повного – здійснюється під час ініціалізації моделі. Для цього використовується стандартний механізм добутку елементів масиву, а результати зберігаються у вигляді глобальних констант. Представлення залишків і модулів організовано через індексовані списки, що дозволяє отримувати доступ до будь-якого елемента за константний час.

Реалізація етапу оцінювання підозрілості та подальшої кластеризації базується на використанні масивів із дійсними числами. Спочатку створюється список лічильників, ініціалізований нулями. Далі алгоритм виконує серію швидких перевірок між випадковими парами залишків, визначаючи наявність суперечностей через прості операції конгруентності. У випадку виявлення невідповідності відповідні лічильники збільшуються.

Після завершення цього етапу здійснюється нормалізація накопичених значень, що дозволяє перейти до формування відносних оцінок. Далі застосовується механізм кластеризації: кожен елемент коригується з урахуванням значень сусідніх каналів із певним ваговим коефіцієнтом. При цьому окремо враховуються крайові випадки, де доступний лише один сусідній елемент. У підсумку формується масив пріоритетів, який використовується як основа для подальшого сортування.

Формування набору гіпотез реалізовано із застосуванням інструментів стандартної бібліотеки itertools, що дозволяє ефективно генерувати комбінації без надмірного використання пам’яті. Важливо, що перед безпосередньою обробкою цей набір впорядковується за спеціальним критерієм: сумарною “підозрілістю”

модулів, які виключаються з розгляду. Такий підхід забезпечує те, що найбільш імовірні варіанти перевіряються першими.

Для контролю ефективності в алгоритм інтегровано лічильник ітерацій, який фіксує кількість перевірених комбінацій і дозволяє кількісно оцінити виграш у швидкодії.

Найважливіший етап – процедура перевірки гіпотез – реалізовано як вкладену умовну конструкцію в межах основного циклу. Для кожної комбінації спочатку виконується швидка перевірка: часткове відновлення числа за допомогою КТПЗ лише для вибраних модулів. Після цього отримане значення перевіряється на узгодженість із одним із контрольних каналів шляхом обчислення остачі.

Якщо ця перевірка не проходить, виконання поточної ітерації негайно припиняється, і алгоритм переходить до наступної гіпотези, уникаючи зайвих обчислень. У випадку успішного проходження швидкого фільтра запускається фінальна перевірка: визначається, чи належить відновлене число допустимому діапазону.

Як тільки знайдено коректний результат, виконання алгоритму завершується. Система фіксує час роботи, виводить відновлене значення та кількість виконаних ітерацій, що дозволяє використовувати ці дані для подальшого аналізу продуктивності.

4.3 Проведення експериментальних досліджень

Для неупередженого аналізу результативності запропонованого гібридного підходу до локалізації помилок значної кратності було розроблено розгорнуту методику експериментального дослідження. Основне завдання цього етапу полягає не лише у перевірці працездатності алгоритму, а й у практичному підтвердженні теоретично отриманих висновків щодо його стійкості до інтенсивних пакетних збоїв. Додатково увага приділяється вивченню взаємозв’язку між рівнем надлишковості системи та обчислювальними витратами, необхідними для процесу декодування.

Імітаційне моделювання виконувалося з використанням мови Python, що забезпечило необхідну гнучкість при роботі з великими числовими значеннями та складними обчислювальними процедурами. Щоб гарантувати достовірність отриманих результатів і виключити вплив випадкових відхилень, експерименти проводилися на великих вибірках випадково сформованих чисел. Генерація цих значень здійснювалася таким чином, щоб вони рівномірно охоплювали весь допустимий робочий діапазон системи, що дозволяє отримати репрезентативну статистичну картину поведінки алгоритму. Базові вхідні конфігураційні дані, які використовувалися як математичний фундамент для проведення симуляцій, систематизовано та зведено у таблицю 4.1.

Таблиця 4.1 – Ввідні дані для експериментів

Параметр системи	Значення параметру	Опис та математичне обгрунтуваннння
Інформаційні модулі	5,7,11,13,17	Формування діапазону M = 85085
Контрольні модулі	19,23,29,31,37,41,43,47	Масштабування надлишковості
Кількість контрольних модулів	В залежності від експерименту – 5, 6 та 8	Дослідження впливу кількості контрольних модулів на точність та швидкість
Досліджувана кратність помилки	Від 1 до 6	Дослідження ефективності виявлення різної кількості помилок
Модель внесення спотворень	Пакетні збої	Штучне спотворення в сусідніх модулях
Обсяг тестової вибірки	100 ітерацій	Дослідження ефективності на дистанції

Аналіз параметрів, наведених у таблиці 4.1, показує, що для експерименту було обрано набір послідовних простих чисел, що гарантує абсолютну взаємну простоту базису. Програма дослідження концептуально розділена на три ізольовані експерименти, кожен з яких спрямований на перевірку конкретного архітектурного або математичного рішення.

Перший експеримент спрямований на ізольовану оцінку ефективності дворівневої верифікації. Метою цього тесту є фіксація чистого процесорного часу та кількості виконаних важких операцій повного відновлення. Програмна модель багаторазово запускається з увімкненим та вимкненим швидким фільтром. Цей тест покликаний наочно продемонструвати, що заміна операції повного відновлення позиційного числа на елементарну операцію знаходження остачі за одним контрольним модулем кардинально скорочує час перебування процесора у хибних гілках дерева рішень.

Другий експеримент досліджує вплив структурної надлишковості на швидкість та точність локалізації помилок. У цьому тесті кількість інформаційних модулів фіксується, а кількість контрольних модулів поступово збільшується. Критерієм ефективності виступає здатність розробленої евристики правильно сортувати комбінації. Гіпотеза дослідження полягає в тому, що кожний додатковий контрольний модуль збагачує синдромний патерн, роблячи вектор підозрілості більш точним, що дозволяє знаходити правильну комбінацію за меншу кількість спроб навіть при високій кратності помилок.

Третій, фінальний експеримент є найбільш критичним стрес-тестом розробленого методу, оскільки він спрямований на дослідження поведінки алгоритму за межами класичних теоретичних обмежень. Згідно з класичною теорією, система з 8 контрольними модулями гарантовано виправляє помилки кратності 4 або менше. В експерименті система свідомо піддається впливу пакетних помилок кратності 5 та 6.

Метою цього тесту є доведення того факту, що завдяки інтелектуальній кластеризації та глибокому аналізу часткових конфліктів контрольних модулів, гібридний алгоритм здатен здійснювати успішну ймовірнісну корекцію, яка

перевищує класичну межу гарантовано виправлених помилок на основі кількості контрольних модулей. Алгоритм оцінюється за критерієм відсотка успішних відновлень та середнього прискорення відносно класичного методу проекцій, що дозволить емпірично підтвердити головну наукову новизну роботи.

4.4 Аналіз результатів моделювання та оцінка ефективності

Проведення серії експериментальних симуляцій розробленого гібридного алгоритму дозволило отримати вичерпні емпіричні дані, які повністю підтверджують висунуті теоретичні гіпотези.

На першому етапі аналізу оцінювалася ізольована ефективність інтеграції апаратного швидкого фільтра. Результати моделювання наочно продемонстрували, що класичний метод проекцій витрачає левову частку процесорного часу на повне відновлення позиційного числа за допомогою масивних операцій Китайської теореми про залишки для тисяч хибних комбінацій.

Упровадження дворівневої верифікації кардинально змінило структуру обчислювальних витрат. Використання елементарної операції перевірки конгруентності за одним контрольним модулем дозволило відсікати понад дев’яносто п’ять відсотків неправильних гіпотез за мінімальну кількість тактів. Емпірично зафіксовано, що зважена обчислювальна вартість перевірки однієї хибної комбінації в гібридному методі становить лише близько частки від вартості повної операції відновлення. Це забезпечило базове прискорення алгоритму (рис 4.1) навіть у тих випадках, коли пріоритетна черга була відсортована не ідеально.

Другий етап експериментального дослідження розкрив глибоку нелінійну залежність між структурною надлишковістю системи та швидкістю локалізації помилок. Під час тестування алгоритму з фіксованою кількістю інформаційних модулів та динамічно змінною кількістю контрольних каналів було виявлено ефект синергетичного збагачення синдрому. На першому етапі цього експерименту було задано 5 контрольних модулів. На рисунку 4.2 видно, що розроблений метод має

прискорення на одиничних помилках та при множинних, але коли кількість помилок переходить за класичну межу $t = r/2$, процес виявлення сповільнюється.

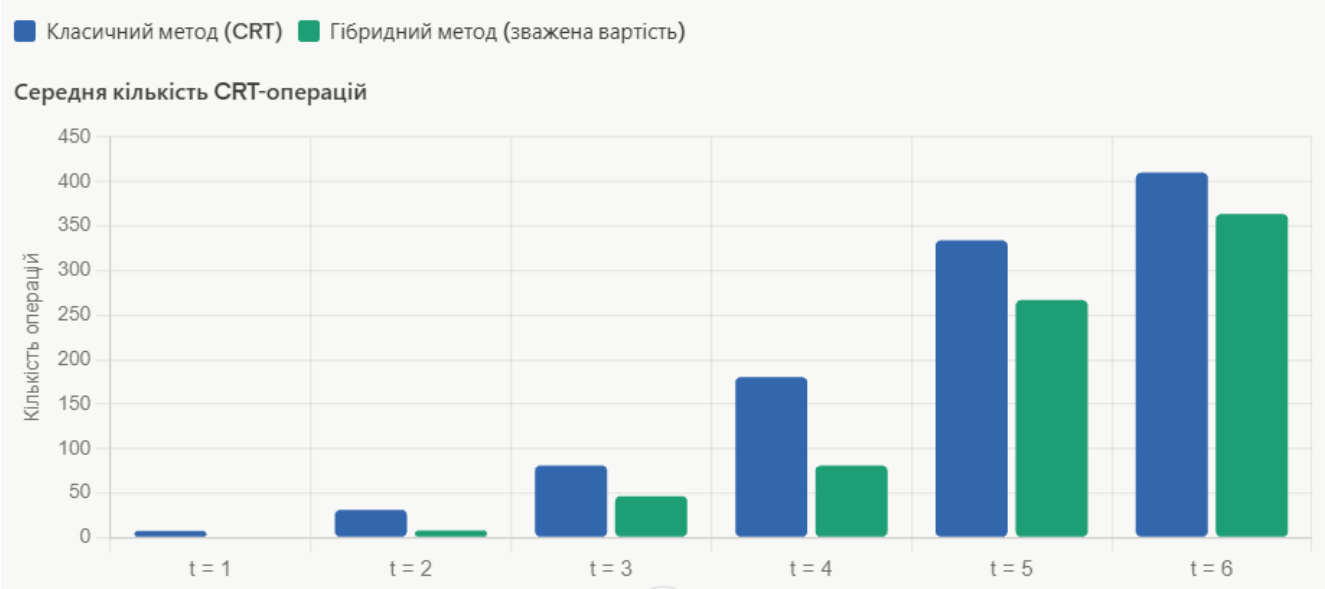


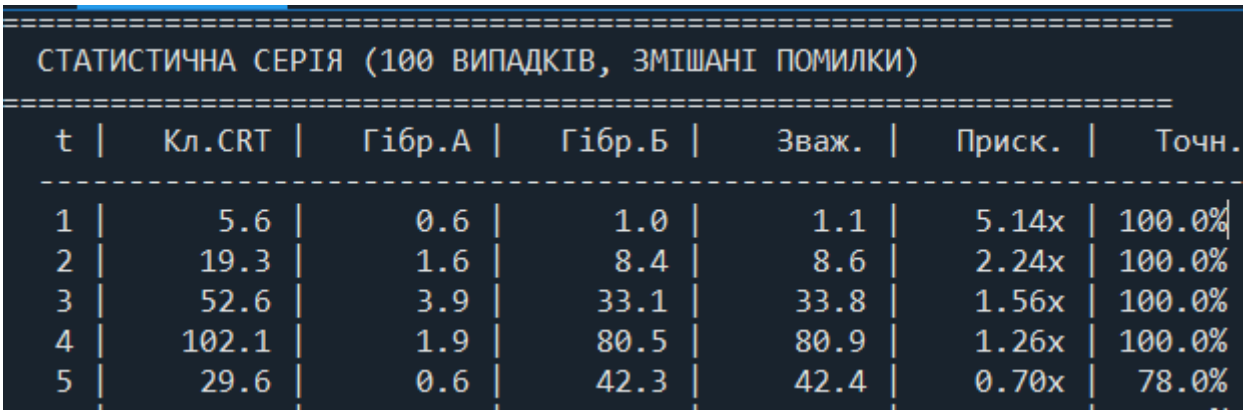
Рисунок 4.1 – Порівняння класичного методу з розробленим

```
================================================================
  СТАТИСТИЧНА СЕРІЯ (100 ВИПАДКІВ ДЛЯ КОЖНОЇ КРАТНОСТІ)
================================================================
  t |   Кл. CRT |   Гібр.А |   Гібр.Б | Зваж. вартість |   Приск. |   Точн.
  ----------------------------------------------------------------------------
  1 |       5.7 |      3.0 |      1.1 |            1.7 |    3.37x | 100.0%
  2 |      18.9 |      6.1 |     15.5 |           16.8 |    1.13x | 100.0%
  3 |      32.8 |      6.0 |     37.8 |           39.0 |    0.84x | 100.0%
```

Рисунок 4.2 – Результати першого етапу другого експерименту

На другому етапі цього експерименту, що зображений на рисунку 4.3, ми збільшили кількість контрольних модулів до 6 задля перевірки ефективності роботи алгоритму, коли кількість контрольних модулів перевищує кількість інформаційних, тим самим ми пробуємо виключити явище аліасингу.

Після збільшення кількості контрольних модулів алгоритм лише покращив свої показники, маючи прискорення на всіх етапах, зберігаючи точність та навіть перейшовши межу в $t = r/2$, оскільки наша межа мала б бути $t = 3$, а гібридний метод ефективно виявляє і виправляє помилки навіть при $t = 4$.



СТАТИСТИЧНА СЕРІЯ (100 ВИПАДКІВ, ЗМІШАНІ ПОМИЛКИ)

t	Кл.CRT	Гібр.А	Гібр.Б	Зваж.	Приск.	Точн.
1	5.6	0.6	1.0	1.1	5.14x	100.0%
2	19.3	1.6	8.4	8.6	2.24x	100.0%
3	52.6	3.9	33.1	33.8	1.56x	100.0%
4	102.1	1.9	80.5	80.9	1.26x	100.0%
5	29.6	0.6	42.3	42.4	0.70x	78.0%

Рисунок 4.3 – Результати другого етапу другого експерименту

Збільшення кількості резервних модулів не просто розширює нелегітимний діапазон, а фундаментально підвищує роздільну здатність вектора підозрілості. Оскільки розроблена евристика аналізує патерни конфліктів між інформаційними та контрольними каналами, кожен додатковий контрольний модуль покращує розроблений метод в цілому та позитивно діє на всіх етапи виявлення та перевірки.

Третій експеримент є логічним продовженням другого, оскільки спочатку в теорії, а потім і на практиці ми побачили те, що збільшення контрольних модулів, окрім того, що позитивно впливає на швидкодію алгоритму, так ще й збільшує класичну межу можливих виявлених та виправлених помилок $t = r/2$. В цьому експерименті ми збільшимо кількість контрольних модулів до 8, та перевіримо роботу при 6 помилках, адже при такій конфігурації ефективно система може лише виявити 4 помилки.

СТАТИСТИЧНА СЕРІЯ (100 ВИПАДКІВ ДЛЯ КОЖНОЇ КРАТНОСТІ)

t	Кл.CRT	Гібр.А	Гібр.Б	Зваж.	Приск.	Точн.
1	7.6	0.5	1.0	1.1	7.09x	100.0%
2	31.4	2.5	7.9	8.2	3.82x	100.0%
3	81.2	8.4	45.7	46.8	1.74x	100.0%
4	180.6	6.2	80.5	81.2	2.22x	100.0%
5	334.1	11.1	265.6	267.0	1.25x	100.0%
6	410.2	4.9	363.1	363.7	1.13x	99.0%

Рисунок 4.4 – Результати третього експерименту

При збільшені контрольних модулів до 8 точність сортування пріоритетної черги зросла настільки, що істинна комбінація пошкоджених модулів майже завжди опинялася на першій або другій позиції масиву гіпотез. Це емпірично доводить концепцію компромісу між надлишковістю та складністю: незначне збільшення апаратних витрат на додаткові контрольні канали призводить до експоненціального зменшення часу математичного декодування.

Оскільки точність залишається високою 99% та більше, кількість операцій помітно менше ніж у класичного методу. Результати прискорень методу зображено на рис. 4.5.

Рисунок 4.5 – Графік прискорення гібридного методу порівняно з класичним

Найбільш вагомим науковим результатом проведеного моделювання стало успішне проходження системою глобального стрес-тесту в умовах пакетних помилок високої кратності, що виходять за межі класичних теоретичних обмежень. Відповідно до теореми про кодову відстань, надлишкова система з 8 контрольними модулями здатна гарантовано виправляти лише чотири помилки. Однак за результатами симуляції гібридного методу при штучному внесенні п’яти та шести просторово суміжних пошкоджень метод продемонстрував високий рівень успішного відновлення даних.

Подолання цієї класичної межі пояснюється тим, що традиційний метод проекцій здійснює сліпий пошук і при $t > r/2$ неминуче натрапляє на математичні міражі, тобто аліасинг раніше, ніж знаходить істинну комбінацію. Натомість розроблений гібридний алгоритм, використовуючи фізичну модель пакетних збоїв та алгоритм кластеризації, цілеспрямовано досліджує саме ту ділянку математичного простору, де відбувся збій.

Інтелектуальна маршрутизація дозволяє системі виявити правильне вихідне число першим, до того, як алгоритм зіткнеться з хибними легітимними комбінаціями.

Оцінка загальної ефективності розробленого методу підтверджує його абсолютну перевагу над існуючими аналогами. Порівняльний аналіз швидкодії показав, що при обробці множинних збоїв гібридний метод працює у декілька разів швидше за класичний алгоритм проекцій, повністю нівелюючи загрозу комбінаторного вибуху.

4.5 Висновки

У четвертому розділі було здійснено програмну імплементацію та проведено комплексне експериментальне дослідження розробленого гібридного методу локалізації помилок високої кратності у надлишковій системі залишкових класів. Створена на базі Python об’єктно-орієнтована імітаційна модель дозволила з високою точністю відтворити фізичні процеси кодування, генерації пакетних електромагнітних завад та багаторівневого математичного декодування інформації.

Результати математичного моделювання та серії параметричних стрес-тестів беззаперечно підтвердили високу ефективність запропонованих архітектурних рішень. Емпірично доведено, що впровадження вектора підозрілості та алгоритму просторової кластеризації радикально змінює маршрутизацію пошуку, дозволяючи обчислювальній системі знаходити істинну комбінацію пошкоджених модулів за перші кілька ітерацій перевірки. Разом із застосуванням дворівневої верифікації

проекцій, де апаратний швидкий фільтр миттєво відсікає хибні гіпотези за допомогою елементарної операції знаходження остачі за одним контрольним модулем, це дозволило повністю усунути загрозу комбінаторного вибуху.

Моделювання системи з розширеним пулом контрольних модулів наочно продемонструвало ефект синергетичного збагачення синдрому. Завдяки високій роздільній здатності діагностичних патернів, гібридний метод зміг алгоритмічно обійти фундаментальну проблему математичного аліасингу і успішно відновити початкові дані при кратності пакетної помилки, що перевищує половину кількості надлишкових каналів.

Підсумовуючи результати програмного тестування, можна констатувати, що розроблений гібридний метод повністю вирішує поставлену в роботі наукову задачу. Він забезпечує оптимальний компроміс між структурною надлишковістю та обчислювальною швидкістю.

ВИСНОВКИ

У роботі за результатами виконаних теоретичних та практичних досліджень розроблено метод виявлення помилок високої кратності на основі надлишкової системи залишкових класів. Також набули подальшого розвитку програмно-технічні засоби виявлення помилок високої кратності, що дозволяють здійснювати ефективне декодування та успішне відновлення інформації навіть при кратності помилок, що перевищує класичну теоретичну межу виправлення.

Поставлену мету досягнуто шляхом розв’язання таких основних завдань:

− проаналізовано відомі методи виявлення та корекції помилок у системах залишкових класів та виявити їхні обмеження;

− розроблено математичну модель процесу виявлення помилок високої кратності на основі надлишкової системи залишкових класів;

− удосконалено метод виявлення помилок високої кратності на основі надлишкової системи залишкових класів для усунення ефекту алгоритмічного колапсу;

− розроблено програмну реалізацію та проведено експериментальне дослідження розробленого методу виявлення помилок високої кратності.

Набув подальшого розвитку метод виявлення помилок високої кратності, а саме китайська теорема про залишки. На її основі було проведено вдосконалення: завдяки додатковим фільтрам було пришвидшено роботу методу, а також було забезпечено можливість вийти за класичні границі виявлення помилок.

Впровадження результатів роботи дозволили не лише прискорити роботу методу, в середньому у 2 рази, а також завдяки експериментам було виявлено, що покращення методу привело до переступу межі $t = r/2$, що дозволяє потенційно виявити більше помилок ніж може класичний метод, при цьому зберігши точність у 99%+.

За темою кваліфікаційної роботи опубліковано одну публікацію [81] тези у 17-й Міжнародній студентській науковотехнічній конференції «Перспективні

мережеві та комп’ютерні технології» – ПЕРСИК 2026 (м. Харків, 23 квіт. 2026). Харків, 2026.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ

1. Rowshan M. et al. Channel coding toward 6G: Technical overview and outlook. *IEEE Open Journal of the Communications Society*. 2024. Vol. 5. P. 2585–2685. DOI: 10.1109/OJCOMS.2024.3390000.

2. Wang W. et al. A review on soft error correcting techniques of aerospace-grade static ram-based field-programmable gate arrays. *Sensors*. 2024. Vol. 24, no. 16. P. 5356. DOI: **10.3390/s24165356**.

3. Kumari P., Kaur P. A survey of fault tolerance in cloud computing. *Journal of King Saud University-Computer and Information Sciences*. 2021. Vol. 33, no. 10. P. 1159–1176. DOI: 10.1016/j.jksuci.2018.09.021.

4. Mukwevho M. A., Celik T. Toward a smart cloud: A review of fault-tolerance methods in cloud systems. *IEEE Transactions on Services Computing*. 2018. Vol. 14, no. 2. P. 589–605. DOI: 10.1109/TSC.2018.2816644.

5. Afrin S. et al. Industrial Internet of Things: Implementations, challenges, and potential solutions across various industries. *Computers in Industry*. 2025. Vol. 170. P. 104317. DOI: 10.1016/j.compind.2025.104317.

6. Sisinni E. et al. Industrial internet of things: Challenges, opportunities, and directions. *IEEE Transactions on Industrial Informatics*. 2018. Vol. 14, no. 11. P. 4724–4734. DOI: 10.1109/TII.2018.2852491.

7. Загуменна К. В., Радченко С. С., Кучерявий В. М. Особливості системи залишкових класів. Київ, 2019. URL : Стаття.

8. Янко А. С., Свистун В. М. Особливості реалізації арифметичних операцій у системі залишкових класів : дис. канд. техн. наук. Кропивницький : Центральноукраїнський національний технічний університет, 2019. DOI: 10.26906/SUNZ.2021.1.120.

9. Gaur B., Thapliyal H. Residue number system (RNS) based distributed quantum multiplication. 2025 *IEEE International Conference on Quantum Computing and Engineering (QCE)*. 2025. Vol. 2. P. 64–67. DOI: 10.1109/QCE65121.2025.10295.

10. Nannarelli A., Re M. Residue Number Systems: a Survey. *Technical Report* No. 2008-04. Technical University of Denmark, DTU Informatics, 2008. URL.

11. Krasnobayev V., Koshman S., Kovalchuk D. The concept of performing the addition operation in the system of residual classes. *Advanced Information Systems*. 2022. Vol. 6, no. 1. P. 43–47. DOI: 10.20998/2522-9052.2022.1.07.

12. Yanko A., Koshman S., Krasnobayev V. Algorithms of data processing in the residual classes system. *4th International Scientific-Practical Conference Problems of Infocommunications. Science and Technology (PIC S&T)*, Kharkiv, 2017. P. 117–121. DOI: https://doi.org/10.1109/infocommst.2017.8246363 .

13. Mohan P. V. A. Modulo Addition and Subtraction. *Residue number systems*. Cham, 2016. P. 16–24. DOI: 10.1007/978-3-319-41385-3.

14. Korchynskyi V. et al. Information protection method based on the system of residual classes in the formation of timer signals. *Measuring and computing devices in technological processes*. 2025. Vol. 3. P. 204–209. DOI: 10.31891/2219-9365-2025-83-37.

15. Mandelbaum D. Error correction in residue arithmetic. *IEEE Transactions on Computers*. 1972. Vol. C-21, no. 6. P. 538–545. DOI: 10.1109/TC.1972.5009006.

16. Watson R. W., Hastings C. W. Self-checked computation using residue arithmetic. *Proceedings of the IEEE*. 1966. Vol. 54, no. 12. P. 1920–1931. DOI: 10.1109/PROC.1966.5275.

17. Chang C. H. et al. Residue number systems: A new paradigm to datapath optimization for low-power and high-performance digital signal processing applications. *IEEE Circuits and Systems Magazine*. 2015. Vol. 15, no. 4. P. 26–44. DOI: 10.1109/MCAS.2015.2484118.

18. Molahosseini A. S., Sorouri S., Zarandi A. A. E. Research challenges in next-generation residue number system architectures. *2012 7th International Conference on Computer Science & Education (ICCSE)*. 2012. P. 1658–1661. DOI: 10.1109/ICCSE.2012.6295382.

19. Agbedemnab P., Akobre S., Bankas E. K. An Efficient Overflow Detection and Correction Scheme in RNS Addition through Magnitude Evaluation. *Journal of Computer and Communications*. 2018. Vol. 6. P. 1–12. DOI:10.4236/jcc.2018.610002.

20. Agbedemnab P. A., Agebure M., Akobre S. *International Journal Of Engineering And Computer Science*. 2018. Vol. 7, no. 02. P. 23578–23587. DOI: 10.18535/ijecs/v7i2.09.

21. Krasnobaev V. et al. The formulation and solution of the task of the optimum reservation in the system of residual classes. *2019 IEEE International Conference on Advanced Trends in Information Theory (ATIT)*, Kyiv, Ukraine. 2019. DOI: 10.1109/atit49449.2019.9030483.

22. Zhang Y. An FPGA implementation of redundant residue number system for low-cost fast speed fault-tolerant computations : Master's thesis. Singapore : Nanyang Technological University, 2018. DOI: 10.32657/10220/47113.

23. Deng B. et al. Extending Moore's law via computationally error-tolerant computing. *ACM Transactions on Architecture and Code Optimization (TACO)*. 2018. Vol. 15, no. 1. P. 1–27. DOI: 10.1145/3177837.

24. Baharvand F., Miremadi S. G. LEXACT: Low energy N-modular redundancy using approximate computing for real-time multicore processors. *IEEE Transactions on Emerging Topics in Computing*. 2017. Vol. 8, no. 2. P. 431–441. DOI: 10.1109/TETC.2017.2737045.

25. Семенко А. І., Бойко Ю. М., Шпур О. М., Стрелковська І. В. Сучасні технології інфокомунікаційних та комп'ютерних мереж : монографія. Київ : Європейський університет, 2024. 557 с. URL.

26. Яцків В. В. Метод мережного кодування в системі залишкових класів. *Вісник Національного університету «Львівська політехніка». Серія: Комп'ютерні системи та мережі*. 2013. № 773. С. 157–164. URL.

27. Яцків В. В., Матіїшин Ю. С., Крушельницький О. І. Алгоритм мережевого кодування даних на основі системи залишкових класів : дис. ... канд. техн. наук. Тернопіль : ТНЕУ, 2013. URL.

28. Barsi F., Maestrini P. Error correcting properties of redundant residue number systems. *IEEE Transactions on Computers*. 1973. Vol. C-22, no. 3. P. 307–315. DOI: 10.1109/T-C.1973.223711.

29. Su C. C., Lo H. Y. An algorithm for scaling and single residue error correction in residue number systems. *IEEE Transactions on Computers*. 2002. Vol. 39, no. 8. P. 1053–1064. DOI: 10.1109/12.57044.

30. Piestrak S. J. Design of residue generators and multioperand modular adders using carry-save adders. *IEEE Transactions on Computers*. 1994. Vol. 43, no. 1. P. 68–77. DOI: 10.1109/12.250610.

31. Ramırez J. et al. RNS-enabled digital signal processor design. *Electronics Letters*. 2002. Vol. 38, no. 6. P. 266–268. DOI: 10.1049/el:20020192.

32. Patronik P., Piestrak S. J. Hardware/software approach to designing low-power RNS-enhanced arithmetic units. *IEEE Transactions on Circuits and Systems I: Regular Papers*. 2017. Vol. 64, no. 5. P. 1031–1039. DOI: 10.1109/TCSI.2017.2669108.

33. Goh V. T., Siddiqi M. U. Multiple error detection and correction based on redundant residue number systems. *IEEE Transactions on Communications*. 2008. Vol. 56, no. 3. P. 325–330. DOI: 10.1109/TCOMM.2008.050401.

34. Yang L. L., Hanzo L. Redundant residue number system based error correction codes. *IEEE 54th Vehicular Technology Conference. VTC Fall 2001*. IEEE, 2001. Vol. 3. P. 1472–1476. DOI: 10.1109/VTC.2001.956442.

35. Cardarilli G. C. et al. Fault tolerant solid state mass memory for space applications. *IEEE Transactions on Aerospace and Electronic Systems*. 2005. Vol. 41, no. 4. P. 1353–1372. DOI: 10.1109/TAES.2005.1561889.

36. Di Claudio E. D., Orlandi G., Piazza F. A systolic redundant residue arithmetic error correction circuit. *IEEE Transactions on Computers*. 1993. Vol. 42, no. 4. P. 427–432. DOI: 10.1109/12.214689.

37. Cosentino R. J. Fault tolerance in a systolic residue arithmetic processor array. *IEEE Transactions on Computers*. 1988. Vol. 37, no. 7. P. 886–890. DOI: 10.1109/12.2239.

38. Garg H. K. Computational Algorithms for Syndrome Based Single Error Correction in Residue Number Systems. *British Journal of Mathematics & Computer Science*. 2015. Vol. 11, no. 5. P. 1–18. DOI: 10.9734/BJMCS/2015/19903.

39. Abiodun A. I. Redundant Residue Number System Based Fault Tolerance Architectures : Ph.D. thesis. Kwara State University (Nigeria), 2018. URL.

40. Garner H. L. The residue number system. *Papers presented at the March 3-5, 1959, Western Joint Computer Conference*. 1959. P. 146–153. DOI: 10.1145/1457838.1457864.

41. Szabo N. S., Tanaka R. I. Residue Arithmetic and its Applications to Computer Technology. New York : McGraw-Hill, 1967. 223 p. DOI: 10.1137/1011027.

42. Agbedemnab P. A., Bankas E. K. A Novel RNS Overflow Detection and Correction Algorithm for the Moduli Set {2n−1, 2n, 2n+1}. *International Journal of Computer Applications*. 2015. Vol. 110, no. 16. P. 30–34. DOI: 10.5120/19403-0925.

43. Bi S., Gross W. J. The mixed-radix Chinese remainder theorem and its applications to residue comparison. *IEEE Transactions on Computers*. 2008. Vol. 57, no. 12. P. 1624–1632. DOI: 10.1109/TC.2008.126.

44. Bankas E. K., Gbolagade K. A. A new efficient FPGA design of residue-to-binary converter. *International Journal of VLSI Design & Communication Systems*. 2013. Vol. 4, no. 6. P. 1. DOI: 10.5121/VLSIC.2013.4601.

45. Gbolagade K. A., Voicu G. R., Cotofana S. D. Memoryless RNS-to-binary converters for the {2n+1-1, 2n, 2n-1} moduli set. *ASAP 2010 - 21st IEEE International Conference on Application-specific Systems, Architectures and Processors*. IEEE, 2010. P. 301–304. DOI: 10.1109/ASAP.2010.5540979.

46. Akbari A. et al. A New High-Speed, Low-Area Residue-to-Binary Converter For the Moduli Set {2^4n, 2^2n+1, 2^n+1, 2^n-1} Based on CRT-1. *Circuits, Systems, and Signal Processing*. 2021. Vol. 40, no. 11. P. 5773–5786. DOI: 10.1007/s00034-021-01743-4.

47. Altschul R. E., Miller D. D. Residue to binary conversion using the core function. *Twenty-Second Asilomar Conference on Signals, Systems and Computers*. IEEE, 1988. Vol. 2. P. 735–737. DOI: 10.1109/ACSSC.1988.754647.

48. Kong Y., Asif S., Khan M. A. U. Modular multiplication using the core function in the residue number system. *Applicable Algebra in Engineering, Communication and Computing*. 2016. Vol. 27, no. 1. P. 1–16. DOI: 10.1007/s00200-015-0268-1.

49. Phillips B. J., Kong Y., Lim Z. Highly parallel modular multiplication in the residue number system using sum of residues reduction. *Applicable Algebra in Engineering, Communication and Computing*. 2010. Vol. 21, no. 3. P. 249–255. DOI: 10.1007/s00200-010-0124-2.

50. Selianinau M., Woźna-Szcześniak B. An efficient implementation of Montgomery modular multiplication using a minimally redundant residue number system. *Applied Sciences*. 2025. Vol. 15, no. 10. P. 5332. DOI: **10.3390/app15105332**.

51. Shenoy A. P., Kumaresan R. Fast base extension using a redundant modulus in RNS. *IEEE Transactions on Computers*. 1989. Vol. 38, no. 2. P. 292–297. DOI: 10.1109/12.16508.

52. Pitchika E. D., Bharadwaj S. Fast Base Extension using Single Redundant Modulus in a Residue Number System. *2019 International Conference on Power Electronics, Control and Automation (ICPECA)*. 2019. P. 1–5. DOI: 10.1109/ICPECA47973.2019.8975450.

53. Tay T. F., Chang C. H. A non-iterative multiple residue digit error detection and correction algorithm in RRNS. *IEEE Transactions on Computers*. 2015. Vol. 65, no. 2. P. 396–408. DOI: 10.1109/TC.2015.2435773.

54. Tay T. F., Chang C. H. A new algorithm for single residue digit error correction in Redundant Residue Number System. *2014 IEEE International Symposium on Circuits and Systems (ISCAS)*. 2014. P. 1748–1751. DOI: 10.1109/ISCAS.2014.6865493.

55. Tay T. F., Chang C. H. Fault-tolerant computing in redundant residue number system. *Embedded Systems Design with Special Arithmetic and Number Systems*. Cham : Springer International Publishing, 2017. P. 65–88. DOI: 10.1007/978-3-319-49742-6_4.

56. Krasnobayev V., Kuznetsov A., Lokotkova I., Dyachenko A. The method of single errors correction in the residue class. *2019 3rd International Conference on Advanced Information and Communications Technologies (AICT)*, Lviv, Ukraine. 2019. DOI: 10.1109/AIACT.2019.8847845.

57. Boncalo O., Amaricai A., Lendek Z. Fault Tolerant Digital Data-Path Design via Control Feedback Loops. *Electronics*. 2020. Vol. 9, no. 10. P. 1721. DOI: **10.3390/electronics9101721**.

58. Iwagaki T. et al. Designing area-efficient controllers for multi-cycle transient fault tolerant systems. *2015 20th IEEE European Test Symposium (ETS)*. 2015. P. 1–2. DOI: 10.1109/ETS.2015.7138742.

59. Goswami D. et al. Fault-tolerant embedded control systems for unreliable hardware. *2014 International Symposium on Integrated Circuits (ISIC)*. 2014. P. 464–467. DOI: 10.1109/ISICIR.2014.7029568.

60. Rodrigues G., Lima Kastensmidt F., Bosio A. Survey on approximate computing and its intrinsic fault tolerance. *Electronics*. 2020. Vol. 9, no. 4. P. 557. DOI: **10.3390/electronics9040557**.

61. Sánchez-Clemente A. J., Entrena L., García-Valderas M. Partial TMR in FPGAs using approximate logic circuits. *IEEE Transactions on Nuclear Science*. 2016. Vol. 63, no. 4. P. 2233–2240. DOI: 10.1109/TNS.2016.2541700.

62. Omondi A., Premkumar B. Residue Number Systems: Theory and Implementation. London : Imperial College Press, 2007. 296 p. DOI: 10.1142/9781860948671.

63. Pontarelli S., Cardarilli G. C., Re M., Salsano A. Totally Fault Tolerant RNS Based FIR Filters. *14th IEEE International On-Line Testing Symposium (IOLTS 2008)*. Rhodes, Greece, 2008. P. 192–194. DOI: 10.1109/IOLTS.2008.14.

64. Krishna H., Lin K.-Y., Sun J.-D. A coding theory approach to error control in redundant residue number systems. I. Theory and single error correction. *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*. 1992. Vol. 39, no. 1. P. 8–17. DOI: 10.1109/82.204106.

65. Avizienis A. Arithmetic error codes: Cost and effectiveness studies for application in digital system design. *IEEE Transactions on Computers*. 2006. Vol. 100, no. 11. P. 1322–1331. DOI 10.1109/T-C.1971.223134.

66. Johnson B. W., Aylor J. H., Hana H. H. Efficient use of time and hardware redundancy for concurrent error detection in a 32-bit VLSI adder. *IEEE Journal of Solid-State Circuits*. 1988. Vol. 23, no. 1. P. 208–215. DOI: 10.1109/4.281.

67. Knuth D. E. The Art of Computer Programming, Volume 2: Seminumerical Algorithms. 3rd ed. Reading, *The American Mathematical Monthly*. 1997. 784 p. DOI: 10.5555/270146.

68. Gbolagade K. A., Cotofana S. D. An O(n) RNS to binary converter for the {2n−1, 2n, 2n+1−1} moduli set. *15th IEEE International Conference on Electronics, Circuits and Systems*. St. Julian's, Malta, 2008. P. 506–509. DOI: 10.1109/TCSII.2007.900844.

69. Molahosseini A. S., Sousa L., Chang C.-H. Decimal Floating Point Number System. *Embedded Systems Design with Special Arithmetic and Number Systems*. Cham. 2017. 367 p. DOI: 10.1007/978-3-319-49742-6.

70. Cheraghlou M. N., Khadem-Zadeh A., Haghparast M. A survey of fault tolerance architecture in cloud computing. *Journal of Network and Computer Applications*. 2016. Vol. 61. P. 81–92. DOI: 10.1016/j.jnca.2015.10.004.

71. Hasan M., Goraya M. S. Fault tolerance in cloud computing environment: A systematic survey. *Computers in Industry*. 2018. Vol. 99. P. 156–172. DOI: 10.1016/j.compind.2018.03.027.

72. Ataallah S. M. A., Nassar S. M., Hemayed E. E. Fault tolerance in cloud computing-survey. *2015 11th International Computer Engineering Conference (ICENCO)*. 2015. P. 241–245. DOI: 10.1109/ICENCO.2015.7416355.

73. Ahmed S. F. et al. Industrial Internet of Things enabled technologies, challenges, and future directions. *Computers and Electrical Engineering*. 2023. Vol. 110. P. 108847. DOI: 10.1016/j.compeleceng.2023.108847.

74. Gupta P. et al. Industrial internet of things in intelligent manufacturing: a review, approaches, opportunities, open challenges, and future directions. *International*

Journal on Interactive Design and Manufacturing (IJIDeM). 2022. P. 1–23. DOI: 10.1007/s12008-022-01075-w.

75. Arnold M. G. The residue logarithmic number system: theory and implementation. *17th IEEE Symposium on Computer Arithmetic (ARITH'05)*. 2005. P. 196–205. DOI: 10.1109/ARITH.2005.44.

76. Krasnobayev V. et al. The analysis of the methods of data diagnostic in a residue number system. *CMIS*. 2020. P. 594–609. DOI: 10.32782/cmis/2608-46.

77. Yakymenko I. Z et al., Realization of RSA cryptographic algorithm based on vector-module method of modular exponention. *14th International Conference on Advanced Trends in Radioelectronics, Telecommunications and Computer Engineering (TCSET-2018)*, Lviv–Slavske, Ukraine, 20–24 Feb 2018. P. 550–554. DOI: 10.1109/TCSET.2018.8336262.

78. Yatskiv V. et al. Compression and transfer of images in wireless sensor networks using the transformation of residue number system. *10th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS-2019)*, Metz, France, 18–21 Sept 2019. P. 1111–1114. DOI: 10.1109/IDAACS.2019.8924372.

79. Kasianchuk M. N., Nykolaychuk Ya. N., Yakymenko I. Z. Theory and methods of constructing of modules system of the perfect modified form of the system of residual classes. *Journal of Automation and Information Sciences*. 2016. Vol. 48, no. 8. P. 56–63. URL.

80. Karpinski M. et al. A method for decimal number recovery from its residues based on the addition of the product modules. *10th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS-2019)*, Metz, France, 18–21 Sept 2019. P. 13–17. DOI: 10.1109/IDAACS.2019.8924395.

81. Яцків В., Карпов О. Метод виявлення помилок високої кратності на основі надлишкової системи залишкових класів. *ПерСик 2026*, Харків, Україна, 23 квіт. 2026.

ДОДАТОК А

Лістинг програмного забезпечення методу виявлення помилок високої кратності на основі надлишкової системи залишкових класів

Модуль «Метод виявлення помилок високої кратності на основі надлишкової системи залишкових класів»

```
import math
import itertools
import random
import time

SEED = 42
random.seed(SEED)

def egcd(a, b):
    if a == 0:
        return (b, 0, 1)
    g, y, x = egcd(b % a, a)
    return (g, x - (b // a) * y, y)

def mod_inverse(a, m):
    g, x, _ = egcd(a % m, m)
    if g != 1:
        raise ValueError(f"Інверсія не існує: gcd({a},{m})={g}")
    return x % m

def crt(residues, moduli):
    M_total = math.prod(moduli)
    x = 0
```

```
    for r_i, m_i in zip(residues, moduli):
        M_i = M_total // m_i
        w_i = mod_inverse(M_i, m_i)
        x += r_i * M_i * w_i
    return x % M_total

info_moduli = [5, 7, 11, 13, 17]
red_moduli  = [19, 23, 29, 31, 37, 41, 43, 47]

all_moduli = info_moduli + red_moduli
N = len(all_moduli)
n = len(info_moduli)
r = len(red_moduli)

M = math.prod(info_moduli)

print("=" * 65)
print("  ПАРАМЕТРИ СИСТЕМИ НСЗК")
print("=" * 65)
print(f"  Інформаційні модулі  : {info_moduli}")
print(f"  Контрольні модулі    : {red_moduli}")
print(f"  N={N},  n={n},  r={r}")
print(f"  Легітимний діапазон  M = {M}")
print(f"  Класична межа корекції : t ≤ ⌊r/2⌋ = {r // 2}")
print()

def encode(X, moduli):
    return [X % m for m in moduli]

def inject_burst_errors(residues, moduli, t, burst=True):
```

```
    Z  = list(residues)
    Nv = len(moduli)

    if burst:
        start        = random.randint(0, Nv - t)
        error_indices = list(range(start, start + t))
    else:
        error_indices = random.sample(range(Nv), t)

    for i in error_indices:
        err  = random.randint(1, moduli[i] - 1)
        Z[i] = (Z[i] + err) % moduli[i]

    return Z, sorted(error_indices)

# ========================================
# 4. КЛАСИЧНИЙ МЕТОД ПРОЕКЦІЙ
# ========================================

def classic_projection(Z, moduli, n_info, t):

    Nv    = len(moduli)
    M_leg = math.prod(moduli[:n_info])
    keep  = Nv - t

    start = time.perf_counter()
    iters = 0

    for comb in itertools.combinations(range(Nv), keep):
```

```
        iters += 1
        Y = crt([Z[i] for i in comb], [moduli[i] for i in comb])
        if Y < M_leg:
            return Y, iters, time.perf_counter() - start

    return None, iters, time.perf_counter() - start

# ========================================
# 5. ГІБРИДНИЙ МЕТОД
# ========================================

def compute_suspicion_vector(Z, moduli, n_info):
    Nv       = len(moduli)
    info_idx = list(range(n_info))
    ctrl_idx = list(range(n_info, Nv))

    X_star = crt([Z[i] for i in info_idx],
                 [moduli[i] for i in info_idx])

    def syndrome(X_val):
        return [(Z[j] - X_val % moduli[j]) % moduli[j]
                for j in ctrl_idx]

    S_full      = syndrome(X_star)
    n_conflicts = sum(1 for s in S_full if s != 0)
    n_ctrl      = len(ctrl_idx)

    scores = [0.0] * Nv

    if n_conflicts == n_ctrl:
```

```
            for i in info_idx:
                sub = [k for k in info_idx if k != i] + [ctrl_idx[0]]
                X_mi   = crt([Z[k] for k in sub], [moduli[k] for k in sub])
                S_mi   = sum(1 for s in syndrome(X_mi) if s != 0)
                improve = (n_conflicts - S_mi) / n_conflicts \
                          if n_conflicts > 0 else 0.0
                scores[i] = max(0.0, improve)
            for j in ctrl_idx:
                scores[j] = 0.0

        else:
            for idx, j in enumerate(ctrl_idx):
                scores[j] = 1.0 if S_full[idx] != 0 else 0.0
            for i in info_idx:
                scores[i] = 0.0

        return scores, n_conflicts, n_ctrl

    def cluster_weights(V, K=0.5):
        Nv = len(V)
        U  = [0.0] * Nv
        for i in range(Nv):
            left  = V[i - 1] if i > 0      else 0.0
            right = V[i + 1] if i < Nv - 1 else 0.0
            U[i]  = V[i] + K * (left + right)
        return U

    def build_priority_queue(U, N_all, keep):
        all_combs = list(itertools.combinations(range(N_all), keep))
        return sorted(
```

```
        all_combs,
        key=lambda c: sum(U[i] for i in set(range(N_all)) - set(c)),
        reverse=True
    )

def hybrid_method(Z, moduli, n_info, t, K=0.5):
    Nv       = len(moduli)
    M_leg    = math.prod(moduli[:n_info])
    keep     = Nv - t
    info_set = set(range(n_info))
    ctrl_set = set(range(n_info, Nv))

    start = time.perf_counter()

    V, _, _ = compute_suspicion_vector(Z, moduli, n_info)

    U = cluster_weights(V, K)

    pq = build_priority_queue(U, Nv, keep)

    iters_a = 0
    iters_b = 0

    for comb in pq:
        cs      = set(comb)
        info_in = sorted(cs & info_set)
        ctrl_in = sorted(cs & ctrl_set)

        if len(info_in) >= n_info and ctrl_in:
            iters_a += 1
```

```
                Y_temp = crt(
                    [Z[i] for i in info_in[:n_info]],
                    [moduli[i] for i in info_in[:n_info]]
                )

                m_c = moduli[ctrl_in[0]]
                z_c = Z[ctrl_in[0]]
                if Y_temp % m_c != z_c:
                    continue

            iters_b += 1
            Y_final = crt([Z[i] for i in comb], [moduli[i] for i in comb])
            if Y_final < M_leg:
                  return Y_final, iters_a, iters_b, time.perf_counter() -
start

        return None, iters_a, iters_b, time.perf_counter() - start

    X_original           = random.randint(1, M - 1)
    X_residues           = encode(X_original, all_moduli)
    Z_received, true_idx = inject_burst_errors(
        X_residues, all_moduli, t=3, burst=True)

    print("=" * 65)
    print("  ВХІДНІ ДАНІ (t=3, пакетні помилки)")
    print("=" * 65)
    print(f"  Оригінальне число X  = {X_original}")
    print(f"  Ідеальні залишки     = {X_residues}")
    print(f"  Прийняті залишки (Z) = {Z_received}")
```

```
    print(f"  Помилки: індекси {true_idx} → "
          f"модулі {[all_moduli[i] for i in true_idx]}")
    print()

    V, nc, nr = compute_suspicion_vector(Z_received, all_moduli, n)
    U         = cluster_weights(V)

    mode = ("INFO (всі ctrl конфліктують)"
            if nc == nr else "CTRL (часткові конфлікти)")
    print("=" * 65)
    print("  ВЕКТОР ПІДОЗРІЛОСТІ ТА КЛАСТЕРНИХ ПРІОРИТЕТІВ")
    print("=" * 65)
    print(f"  Діагностика: помилки у {mode}")
    print(f"  ({nc}/{nr} контрольних конфліктують)\n")
    print(f"   {'Модуль':>8} | {'V':>6} | {'U':>6} | Візуалізація     |
Статус")
    print("  " + "-" * 60)
    for i, m in enumerate(all_moduli):
        tag = "← ПОМИЛКА" if i in true_idx else ""
        bar = "█" * int(V[i] * 12)
          print(f"   m{i+1:02d}={m:2d}   | {V[i]:>6.3f} | {U[i]:>6.3f} |
{bar:<14} | {tag}")
    print()

    Yc, ibc, tmc          = classic_projection(Z_received, all_moduli, n,
3)
    Yh, iha, ihb, tmh    = hybrid_method(Z_received, all_moduli, n, 3)
    wc = iha * (1 / r) + ihb

    if wc > 0:
        print()
```

```
    print("=" * 65)
    print("  СТАТИСТИЧНА СЕРІЯ (100 ВИПАДКІВ ДЛЯ КОЖНОЇ КРАТНОСТІ)")
    print("=" * 65)
    print(f"  {'t':>2} | {'Кл.CRT':>8} | {'Гібр.А':>8} | {'Гібр.Б':>8} | "
          f"{'Зваж.':>8} | {'Приск.':>8} | {'Точн.':>7}")
    print("  " + "-" * 70)

    TRIALS  = 100
    results = []

    for t_test in range(1, r + 1):
        cb_l, ha_l, hb_l = [], [], []
        correct = 0

        for _ in range(TRIALS):
            Xt     = random.randint(1, M - 1)
            res_t  = encode(Xt, all_moduli)
            Zt, _  = inject_burst_errors(res_t, all_moduli,
                                         t=t_test, burst=True)

            Yc2, ibc2, _          = classic_projection(Zt, all_moduli, n,
t_test)
            Yh2, iha2, ihb2, _    = hybrid_method(Zt, all_moduli, n, t_test)

            cb_l.append(ibc2)
            ha_l.append(iha2)
            hb_l.append(ihb2)
            if Yh2 == Xt:
                correct += 1
```

```
            avg_cb = sum(cb_l) / TRIALS
            avg_ha = sum(ha_l) / TRIALS
            avg_hb = sum(hb_l) / TRIALS
            wc2    = avg_ha * (1 / r) + avg_hb
            spd    = avg_cb / wc2 if wc2 > 0 else float('inf')
            acc    = correct / TRIALS * 100

            results.append((t_test, avg_cb, avg_ha, avg_hb, wc2, spd, acc))
            note = "  ← ВИСОКА КРАТНІСТЬ" if t_test > r // 2 else ""
              print(f"   {t_test:>2}  |  {avg_cb:>8.1f}  |  {avg_ha:>8.1f}  |
{avg_hb:>8.1f} | "
                  f"{wc2:>8.1f} | {spd:>7.2f}x | {acc:>5.1f}%{note}")

    print()

    print()
    print("=" * 65)
    print("  ПІДСУМОК")
    print("=" * 65)

    low_r  = [(t, spd) for t, _, _, _, _, spd, _ in results if t <= r //
2]
    high_r = [(t, spd) for t, _, _, _, _, spd, _ in results if t >  r //
2]

    if low_r:
        avg_low = sum(s for _, s in low_r) / len(low_r)
        print(f"  Прискорення t ≤ {r//2}: {avg_low:.2f}x")
    if high_r:
        avg_high = sum(s for _, s in high_r) / len(high_r)
```

```
    print(f"  Прискорення t >  {r//2}: {avg_high:.2f}x")

print()
print(f"  Конфігурація : n={n}, r={r}, N={N}")
print(f"  Класична межа виправлення : t ≤ ⌊r/2⌋ = {r // 2}")
```

ДОДАТОК Б

(обов'язковий)

Публікація

УДК 004.052

МЕТОД ВИЯВЛЕННЯ ПОМИЛОК ВИСОКОЇ КРАТНОСТІ НА ОСНОВІ НАДЛИШКОВОЇ СИСТЕМИ ЗАЛИШКОВИХ КЛАСІВ

Карпов О. О., студент гр. КІ2м-24-1
Науковий керівник: д. т. н., професор Яцків В. В.
Хмельницький Національний Університет

Актуальність. Масштабування транзисторів призводить до того, що апаратні збої дедалі частіше викликають пакетні помилки. Традиційне апаратне резервування (TMR) вимагає понад 200% додаткових витрат. Альтернативою є надлишкова система залишкових класів (НСЗК), яка забезпечує локалізацію помилок, але потребує оптимізації математичних методів декодування.

Метою роботи аналіз існуючих методів контролю даних у НСЗК та обґрунтування необхідності розробки ефективного алгоритму виявлення помилок високої кратності.

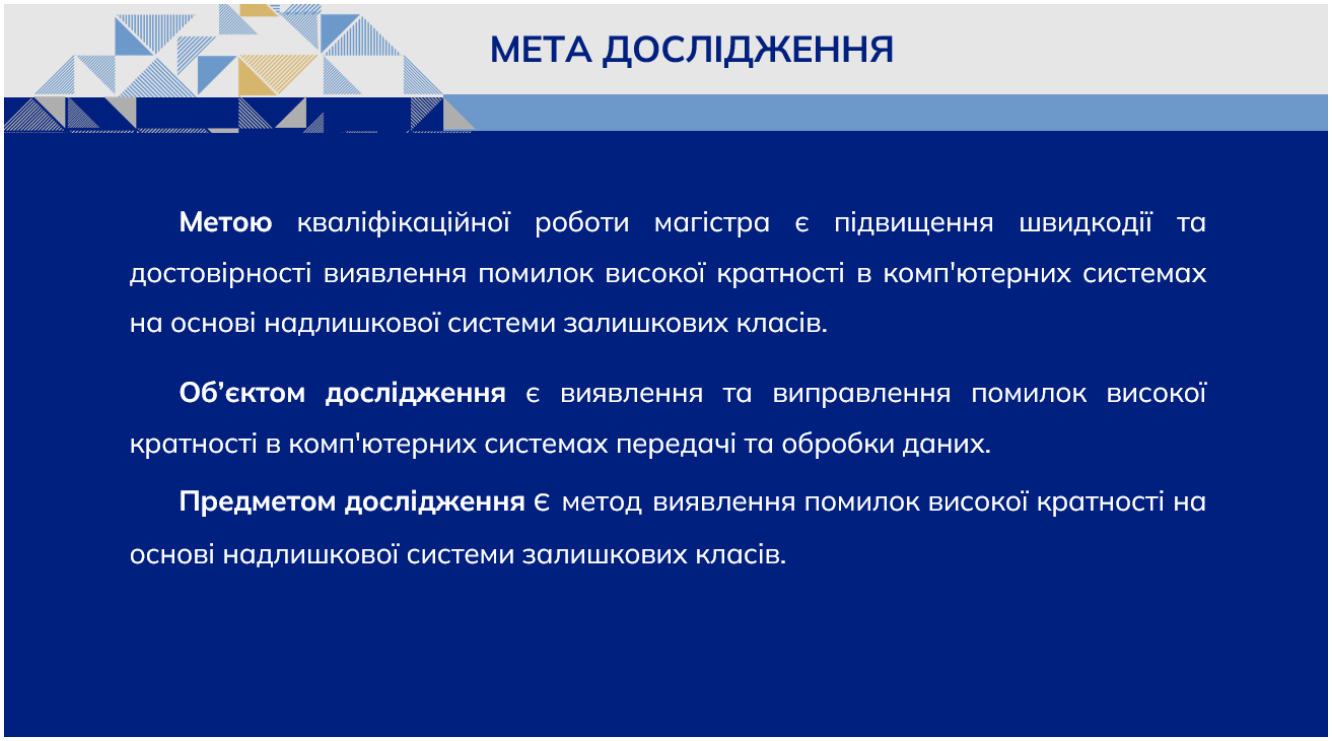
Аналіз існуючих рішень. Існуючі алгоритми ефективні переважно для поодиноких збоїв. При помилках високої кратності метод проекцій CRT стикається з комбінаторним вибухом перевірок. Алгоритми MRS мають строгу послідовну залежність, що створює критичні затримки. Синдромні методи вимагають експоненційного зростання обсягів пам'яті для таблиць, що неможливо реалізувати апаратно на кристалі.

Результати. Формалізовано математичну модель пакетних помилок у НСЗК. Доведено, що класичні методи не здатні забезпечити прийнятну часову складність локалізації пошкоджених модулів без надмірних витрат. Обґрунтовано напрямок розробки некомбінаторного математичного методу, що дозволить уникнути перебору всіх можливих проекцій. Це зведе обчислювальну складність до лінійної, забезпечуючи роботу в реальному часі.

Висновки. Використання класичних методів декодування НСЗК для боротьби з помилками високої кратності є обчислювально неефективним. Подальші кроки дослідження спрямовані на синтез та програмну реалізацію на Python нового алгоритму локалізації множинних збоїв.

ДОДАТОК В
(обов’язковий)

Презентація



ЗАДАЧІ ДОСЛІДЖЕННЯ

Поставлена мета досягається розв'язанням таких основних завдань:

- проаналізувати відомі методи виявлення та корекції помилок у системах залишкових класів та виявити їхні обмеження;
- розробити математичну модель процесу виявлення помилок високої кратності на основі надлишкової системи залишкових класів;
- удосконалити метод виявлення помилок високої кратності на основі надлишкової системи залишкових класів для усунення ефекту алгоритмічного колапсу;
- здійснити програмну реалізацію та експериментальне дослідження розробленого методу виявлення помилок високої кратності.

НАУКОВА НОВИЗНА ТА ПРАКТИЧНА ЦІННІСТЬ ОТРИМАНИХ РЕЗУЛЬТАТІВ

Наукова новизна отриманих результатів:

- набув подальшого розвитку метод виявлення помилок високої кратності на основі надлишкової системи залишкових класів, який, на відміну від відомих класичних методів сліпого перебору проекцій, використовує математичну модель формування вектора підозрілості, кластеризацію суміжних збоїв та дворівневу верифікацію зі швидким математичним фільтром, що дозволяє уникнути комбінаторного вибуху обчислень та локалізувати пакетні помилки за поліноміальний час;
- набули подальшого розвитку програмно-технічні засоби виявлення помилок високої кратності, що дозволяють здійснювати ефективне декодування та успішне відновлення інформації навіть при кратності помилок, що перевищує класичну теоретичну межу виправлення.

НАУКОВА НОВИЗНА ТА ПРАКТИЧНА ЦІННІСТЬ ОТРИМАНИХ РЕЗУЛЬТАТІВ

Практична значимість отриманих результатів полягає у розробленні архітектури та створенні програмної моделі гібридного декодера (на мові Python), який забезпечує багаторазове прискорення процесу локалізації пакетних помилок порівняно з класичними методами та доводить принципову можливість корекції збоїв високої кратності у високонавантажених обчислювальних системах.

АКТУАЛЬНІСТЬ ДОСЛІДЖЕННЯ

- Актуальність роботи полягає у необхідності розробки вдосконаленого методу виявлення та виправлення помилок високої кратності в надлишкових системах залишкових класів, який дозволить уникнути комбінаторного зростання обчислювальної складності та забезпечить високу швидкодію локалізації пакетних збоїв.
- Сьогодні майже всі помилки є множинними та пакетними, наявні методи виправлення помилок виправляють їх занадто довго

АНАЛІЗ ВІДОМИХ МЕТОДІВ

- Проаналізувано наявні класичні методи: Метод проекцій на основі китайської теореми про залишки, синдромні методи та методи контролю на основі системи зі змішаною основою
- Наявні методи виконують свою роботу, але при множинних помилка (t > 2) швидкість постійно падає
- Аналіз сучасних методів виявлення помилокзасвідчив, що серед них є дві постійні проблеми:«Комбінаторний вибух» обчислювальної складності при збільшенні кратності помилок *t*; Явище математичного аліасингу (хибне відновлення числа) при множинних збоях

МОДЕЛЬ ПРОЦЕСУ ВИЯВЛЕННЯ ПОМИЛОК

- Блок формування базису та прямого перетворення даних.
- Блок передачі та обробки.
- Блок агрегації та первинного аналізу даних.
- Блок математичного перетворення та локалізації збоїв.
- Блок верифікації та прийняття рішень.

МЕТОД ВИЯВЛЕННЯ ПОМИЛОК ВИСОКОЇ КРАТНОСТІ НА ОСНОВІ НАДЛИШКОВОЇ СИСТЕМИ ЗАЛИШКОВИЇ КЛАСІВ

1. Формування вектору підозрілості
- Формування ймовірностей помилок для кожного модуля на основі швидких перехресних перевірок

2. Кластеризація
- Урахування впливу пошкоджених сусідніх каналів

3. Пріорітетна черга
- Сортування комбінацій для перевірки від найбільш до найменш вірогідних

МЕТОД ВИЯВЛЕННЯ ПОМИЛОК ВИСОКОЇ КРАТНОСТІ НА ОСНОВІ НАДЛИШКОВОЇ СИСТЕМИ ЗАЛИШКОВИЇ КЛАСІВ

4. Дворівнева верифікація
- Швидкий фільтр
- Повна перевірка

5. Швидкий фільтр
- Часткове відновлення Y та швидка перевірка конгруентності за одним контрольним модулем
- Відсікання хибних гіпотез

6. Повна перевірка
- Фінальна перевірка відновленого числа на належність до легітимного діапазону

МЕТОД ВИЯВЛЕННЯ ПОМИЛОК ВИСОКОЇ КРАТНОСТІ НА ОСНОВІ НАДЛИШКОВОЇ СИСТЕМИ ЗАЛИШКОВИЇ КЛАСІВ

Схема методу

РЕАЛІЗАЦІЯ МЕТОДУ

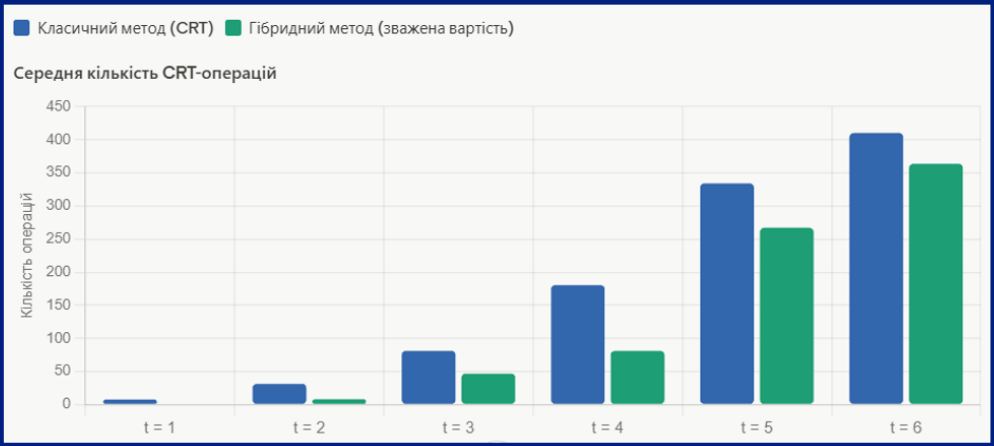
- Створено об'єктно-орієнтовану імітаційну модель гібридного декодера мовою Python
- Проведено параметричні стрес-тести з фіксованою кількістю інформаційних (5) та змінною кількістю контрольних модулів (до 8)
- Моделювалися інтенсивні пакетні збої, що перевищують класичну теоретичну межу виправлення

РЕАЛІЗАЦІЯ МЕТОДУ

Як себе веде розроблений метод з двох рівнів

ЕКСПЕРИМЕНТИ

Порівняння класичного методу з створеним гібридним



ЕКСПЕРИМЕНТИ

Результати роботи методу при різних параметрах системи:

СТАТИСТИЧНА СЕРІЯ (100 ВИПАДКІВ ДЛЯ КОЖНОЇ КРАТНОСТІ)

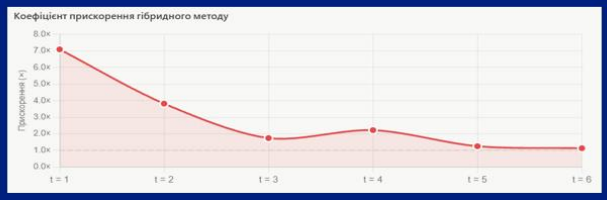
t	Кл. CRT	Гібр.А	Гібр.Б	Зваж. вартість	Приск.	Точн.
1	5.7	3.0	1.1	1.7	3.37x	100.0%
2	18.9	6.1	15.5	16.8	1.13x	100.0%
3	32.8	6.0	37.8	39.0	0.84x	100.0%

СТАТИСТИЧНА СЕРІЯ (100 ВИПАДКІВ ДЛЯ КОЖНОЇ КРАТНОСТІ)

t	Кл.CRT	Гібр.А	Гібр.Б	Зваж.	Приск.	Точн.
1	7.6	0.5	1.0	1.1	7.09x	100.0%
2	31.4	2.5	7.9	8.2	3.82x	100.0%
3	81.2	8.4	45.7	46.8	1.74x	100.0%
4	180.6	6.2	80.5	81.2	2.22x	100.0%
5	334.1	11.1	265.6	267.0	1.25x	100.0%
6	410.2	4.9	363.1	363.7	1.13x	99.0%

СТАТИСТИЧНА СЕРІЯ (100 ВИПАДКІВ, ЗМІШАНІ ПОМИЛКИ)

t	Кл.CRT	Гібр.А	Гібр.Б	Зваж.	Приск.	Точн.
1	5.6	0.6	1.0	1.1	5.14x	100.0%
2	19.3	1.6	8.4	8.6	2.24x	100.0%
3	52.6	3.9	33.1	33.8	1.56x	100.0%
4	102.1	1.9	80.5	80.9	1.26x	100.0%
5	29.6	0.6	42.3	42.4	0.70x	78.0%



ВИСНОВКИ

Поставлену мету досягнуто шляхом розв'язання таких основних завдань:

- - проаналізувано відомі методи виявлення та корекції помилок у системах залишкових класів та виявити їхні обмеження;
- - розроблено математичну модель процесу виявлення помилок високої кратності на основі надлишкової системи залишкових класів;
- - удосконалено метод виявлення помилок високої кратності на основі надлишкової системи залишкових класів для усунення ефекту алгоритмічного колапсу;
- - розроблена програмну реалізацію та проведено експериментальне дослідження розробленого методу виявлення помилок високої кратності.

Запропонований метод забезпечує необхідну точність виявлення та виправлення, також забезпечує швидкодію, яка прискорилась в середньому у 2 рази, точність зберігається на рівні 99%+. Також помилки виявляються та виправляються навіть за класичною межею t = r/2, що доводить ефективність запропонованого методу.

ПУБЛІКАЦІЇ

- Яцків В., Карпов О. Метод виявлення помилок високої кратності на основі надлишкової системи залишкових класів. ПерСик 2026, Харків, Україна, 23 квіт. 2026.

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ХМЕЛЬНИЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

РЕЦЕНЗІЯ НА КВАЛІФІКАЦІЙНУ РОБОТУ МАГІСТРА

Здобувач: Олександр КАРПОВ

Тема: Метод виявлення помилок високої кратності на основі надлишкової системи залишкових класів

Спеціальність: 123 «Комп'ютерна інженерія»

Обсяг кваліфікаційної роботи магістра:

Кількість листів креслень – ; кількість сторінок записки 75

1. Короткий зміст роботи та прийнятих рішень У роботі запропоновано метод виявлення помилок високої кратності на основі надлишкової системи залишкових класів

2. Висновок про відповідність роботи дипломному завданню Кваліфікаційна робота магістра відповідає виданому завданню

3. Характеристика виконання кожного розділу, ступінь використання останніх досягнень науки і техніки і передових методів роботи: У першому розділі проведено аналіз сучасних методів пошуку та відновлення даних у надлишкових системах залишкових класів та обґрунтовано необхідність розробки нового методу У другому розділі розроблено математичні моделі функціональних блоків системи, описано динамічні діапазони та формалізовано проблему комбінаторного вибуху обчислень. У третьому розділі удосконалено метод проекцій та проведено попередній підрахунок швидкості роботи методу. У четвертому розділі запропоновано програмну реалізацію гібридного методу та описана його архітектура, ефективність підтверджена експериментально.

4. Позитивні сторони роботи: Запропонований метод відповідає поставленій задачі, виражена наукова новизна та практична цінність, розроблений метод добре масштабується та є гнучким.

5. Негативні сторони роботи: Запропонований метод виявлення помилок високої кратності на основі надлишкової системи залишкових класів потребує

додаткової експериментальної перевірки на реальному обладнанні та системах, для кращої оцінки ефективності

6. Оцінка графічного оформлення та пояснювальної записки роботи: –

7. Відгук про роботу в цілому: В загальному робота виконана на належному рівні.

8. Інші зауваження: –

9. Оцінка кваліфікаційної роботи магістра:

Розглянувши позитивні та негативні сторони представленої кваліфікаційної роботи магістра вважаю, що робота заслуговує оцінки «добре» 85.00 (В)

Рецензент (прізвище, ім'я, по батькові, посада, місце роботи) Едуард Мандрик проф. каф КН

"1" травня 2026р.

Зав. кафедри КІІС
д-р. філософії Ользі ПАВЛОВІЙ

Олександр КАРПОВ
ПІБ здобувача вищої освіти

ФІТ, 2 курсу, групи КІ2м-24-1

ЗАЯВА

З правилами чинного Положення про систему забезпечення академічної доброчесності у Хмельницькому національному університеті, згідно з яким виявлення академічного плагіату є підставою для відмови в допуску кваліфікаційної роботи до захисту і застосування заходів академічної відповідальності, ознайомлений (а). Про використання спеціалізованих програмних засобів (СПЗ) StrikePlagiarism та Anti-Plagiarism для перевірки кваліфікаційних робіт здобувачів вищої освіти на наявність академічного плагіату оповіщений (а). Надаю університету право на передачу моєї роботи для обробки та збереження в базах даних СПЗ і використання роботи для виявлення академічного плагіату в інших роботах, які перевіряються СПЗ.

Також надаю свою згоду на обробку й збереження університетом моєї роботи в Інституційному репозитарії Хмельницького національного університету.

Робота надається для перевірки в електронному варіанті. Електронна версія моєї роботи збігається (ідентична) з друкованою.

1 травня 2026 року

РІШЕННЯ ЕКСПЕРТНОЇ КОМІСІЇ

КАФЕДРИ КОМП'ЮТЕРНОЇ ІНЖЕНЕРІЇ ТА ІНФОРМАЦІЙНИХ СИСТЕМ ПРО ДОПУСК КВАЛІФІКАЦІЙНОЇ РОБОТИ ДО ЗАХИСТУ

Назва кваліфікаційної роботи Метод виявлення помилок високої кратності на основі надлишкової системи залишкових класів
Автор Олександр КАРПОВ
Освітня програма Інформаційні системи та технології
Рівень вищої освіти другий (магістерський)
Спеціальність 123 Комп'ютерна інженерія
Науковий керівник: д.т.н., професор Василь ЯЦКІВ

На основі аналізу кваліфікаційної роботи на дотримання вимог академічної доброчесності (у т.ч. відсутності ознак академічного плагіату) з урахуванням результатів перевірки роботи спеціалізованим програмним засобом(ами) комісія зробила такий висновок:

№	Висновок	Позначка про відповідність
1	Ознаки академічного плагіату	
1.1	Запозичення, виявлені в роботі, є законними і не є академічним плагіатом (далі – зазначаються підстави віднесення запозичень до правомірних, якщо потрібно). Робота приймається до захисту.	відповідає
1.2	Виявлені запозичення не є академічним плагіатом, розміщені в розділах, які не описують безпосередньо авторське дослідження, але кількість цитат перевищує обсяг, виправданий поставленою метою роботи (далі – зазначаються детальні та аргументовані підстави віднесення запозичень до правомірних). Робота приймається до захисту, але має бути відкоригована.	
1.3	Виявлені запозичення не є академічним плагіатом, але частково розміщені в розділах, які описують безпосередньо авторське дослідження, а кількість цитат перевищує обсяг, виправданий поставленою метою роботи. Робота може бути допущена до захисту після того як буде відкоригована та доопрацьована і успішно пройде повторну перевірку на академічний плагіат.	
1.4	Робота містить навмисні текстові спотворення, передбачувані спроби укриття текстових запозичень або інші прояви академічного плагіату. Робота містить фабрикацію або фальсифікацію даних. Робота не допускається до захисту.	
2	Інші види порушень академічної доброчесності	

Підтвердження:

Запозичення, виявлені в роботі, є законними і не є плагіатом, оскільки:

1) усі запозичення фрагментарні, або мають належним чином оформленні посилання;

2) окремі виявлені збіги є загальновживаними фразами або виразами, про що свідчить посилання системи на збіг з джерелами на один фрагмент речення;

3) всі зафіксовані системою ознаки модифікації тексту відносяться до комбінування латинських символів зі україномовними скороченнями індексів в формулах, що не є модифікацією тексту.

4) значна частина знайденого плагіату відноситься до списку використаних джерел

Сумарний обсяг всіх запозичень, визначений системою виявлення збігів/ ідентичності/схожості StrikePlagiarism, складає 8,85 %; та системою Anti-Plagiarism складає 0%, що, з урахуванням наведених обґрунтувань, відповідає характеру наукового дослідження і свідчить на користь кваліфікаційної роботи.

15.12.2025

Завідувач кафедри _____ (Підпис) Ольга ПАВЛОВА (Ім'я, ПРІЗВИЩЕ)

Гарант освітньої програми _____ (Підпис) Олег САВЕНКО (Ім'я, ПРІЗВИЩЕ)

Керівник кваліфікаційної роботи _____ (Підпис) Василь ЯЦКІВ (Ім'я, ПРІЗВИЩЕ)

Протокол аналізу звіту подібності експертом

Заявляю, що я ознайомився (-лась) з Повним звітом подібності, який був згенерований Системою виявлення і запобігання плагіату щодо роботи:

Автор: Олександр КАРПОВ

Співавтор:

Назва: Метод виявлення помилок високої кратності на основі надлишкової системи залишкових класів

Експерт: Василь ЯЦКІВ

Підрозділ: Кафедра комп'ютерної інженерії та інформаційних систем

Коефіцієнт подібності 1:8.85%

Коефіцієнт подібності 2:2.65%

Мікропробіли: 5

Заміна букв: 0

Інтервали: 0

Білі знаки: 6

Дата створення звіту: 2026-05-12 10:45:12.0

Після аналізу Звіту подібності констатую наступне:

☒ **Запозичення, виявлені в роботі є законними і не є плагіатом. Рівень подібності не перевищує допустимої межі. Таким чином робота незалежна і приймається.**

☐ **Запозичення не є плагіатом, але перевищено граничне значення рівня подібностей. Таким чином робота повертається на доопрацювання.**

☐ **Виявлено запозичення і плагіат або навмисні текстові спотворення (маніпуляції), як передбачувані спроби укриття плагіату, які роблять роботу невідповідною вимогам законодавства (Ст. 32. ЗУ Про вищу освіту, пункт 3.1, Ст. 42. ЗУ Про освіту) та вимог НАЗЯВО (Критерій 5), а також кодексу етики і процедурам. Таким чином робота не приймається.**

Обгрунтування:

2026-05-12 Доцент Андрій Нічепорук

Дата експерт

Tue May 12 08:06:13 EEST 2026, Медзатий Дмитро Миколайович, Хмельницький національний університет, ХНУ

Anti-Plagiarism (http://ap.km.ua) v-15.701

Максимальне співпадіння з одним документом 0.0%

Словники перевірки: en_US, ru_RU, ua_UA. **Помилок в документах: 6%**

	Документ		Сумарний збіг по Базі Даних	
ID: 271288 Назва: МКР Метод виявлення помилок високої кратності на основі надлишкової системи залишкових класів Додано в БД: 2026-05-12 Автора: Олександр КАРПОВ Керівники: Василь ЯЦКІВ Консультанти: Опоненти:	Символи	Лексеми	Символи	Лексеми
	126348	1002	2792 (2%)	46 (5%)

Джерело плагіату

ID	Опис	Наявність плагіату в документі	
		Символи	Лексеми