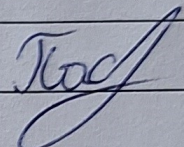


Бакалаврська кваліфікаційна робота на тему:

**«РОЗРОБКА АВТОМАТИЗОВАНОГО МОДУЛЯ ДЛЯ ДЕКОДУВАННЯ
ТУРБО-КОДІВ НА ОСНОВІ ПОШУКУ МАКСИМАЛЬНО
ПРАВДОПОДІБНОЇ БІНАРНОЇ ПОСЛІДОВНОСТІ»**

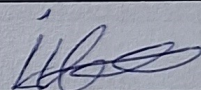
Виконала: студентка IV курсу, групи 1АКІТ-216
спеціальності 151 – Автоматизація та
комп'ютерно-інтегровані технології

Пасічнюк Т. В.



Керівник: к.т.н., доцент каф. АІТ

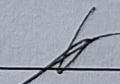
Іванов Ю. Ю.



« 12 » 06 2025 р.

Рецензент: д.т.н., професор каф. КСУ

Ковтун В.В.

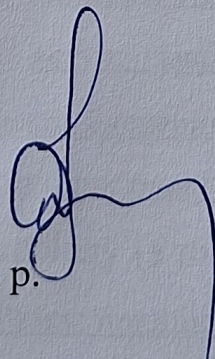


« 12 » 06 2025 р.

Допущено до захисту
зав. кафедри АІТ

д.т.н., проф. Бісікало О. В.

« 13 » 06 2025 р.



Вінницький національний технічний університет

Факультет інтелектуальних інформаційних технологій та автоматизації

Кафедра автоматизації та інтелектуальних інформаційних технологій

Рівень вищої освіти перший (бакалаврський)

Галузь знань 15 – Автоматизація та приладобудування

Спеціальність 151 – Автоматизація та комп'ютерно-інтегровані технології

Освітньо-професійна програма Інтелектуальні комп'ютерні системи управління

ЗАТВЕРДЖУЮ

завідувач кафедри АІТ

д.т.н., проф. Бісікало О. В.

«13» 06 2025 р.

З А В Д А Н Н Я

НА БАКАЛАВРСЬКУ КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТЦІ

Пасічнюк Тетяні Василівні

1. Тема роботи: Розробка автоматизованого модуля для декодування турбо-кодів на основі пошуку максимально правдоподібної бінарної послідовності.
Керівник роботи: к.т.н., доц. каф. АІТ Іванов Юрій Юрійович
Затверджені наказом ВНТУ від «20» 03 2025 року № 97.
2. Термін подання студентом роботи: «до 23» 06 2025 р.
3. Вихідні дані до роботи: згортковий турбо-код; математичний апарат турбо-декодування на основі пошуку максимально правдоподібної бінарної послідовності (розмір пакету n , кількість кодерів v та ітерацій декодування q); послідовності тестових даних.
4. Зміст текстової частини: вступ; аналіз предметної області роботи; розробка математичної моделі турбо-кодека; експериментальні дослідження ефективності роботи турбо-кодека; висновки; список використаних джерел.
5. Перелік ілюстративного матеріалу (з точним зазначенням обов'язкових креслень): структура системи цифрового зв'язку; принцип турбо-декодування; схема програми.

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
1-3	Іванов Ю. Ю., к.т.н., доц. каф. АІТ	21.03.2025 	02.06.2025

7. Дата видачі завдання: «21» 03 2025 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва та зміст етапу	Термін виконання		Примітки
		початок	закінчення	
1	Аналіз предметної області роботи	21.03.2025	04.04.2025	Виконано
2	Розробка математичної моделі турбо-кодека	05.04.2025	26.04.2025	Виконано
3	Експериментальні дослідження ефективності роботи турбо-кодека	27.04.2025	16.05.2025	Виконано
4	Оформлення пояснювальної записки та ілюстративного матеріалу	17.05.2025	27.05.2025	Виконано
5	Попередній захист роботи	28.05.2025	02.06.2025	Виконано
6	Остаточний захист роботи	16.06.2025	23.06.2025	Виконано

Студент

Керівник роботи

(підпис)

(підпис)

Пасічнюк Т. В.
(прізвище та ініціали)
Іванов Ю. Ю.
(прізвище та ініціали)

АНОТАЦІЯ

Бакалаврська кваліфікаційна робота складається з 69 сторінок формату А4, на яких є 21 рисунок, 4 таблиці, список використаних джерел містить 26 найменувань.

У роботі представлено математичний апарат алгоритмів декодування згорткових турбо-кодів на основі пошуку максимально правдоподібної послідовності бінарних символів. Розроблено алгоритмічне та програмне забезпечення, яке дозволяє виконати експериментальні дослідження ефективності роботи турбо-декодера.

Ключові слова: дані, завадостійкість, згортковий турбо-код, передавання даних, кодек, декодер.

ABSTRACT

The bachelor's thesis consists of 69 A4 pages with 21 figures, 4 tables, the list of sources used contains 26 items.

In the given work the mathematical apparatus of the methods for decoding convolutional turbo-codes based on the search for the most likelihood sequence of binary symbols, has been presented. The algorithm and software, that allows to perform experimental researches of the turbo-decoder efficiency, have been developed.

Keywords: data, error correction, convolutional turbo-code, data transmission, codec, decoder.

ЗМІСТ

ВСТУП	3
1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ РОБОТИ.....	6
1.1 Основна термінологія	6
1.2 Методи завадостійкого кодування даних	9
1.3 Дослідження об'єкта автоматизації.....	17
1.4 Висновки	20
2 РОЗРОБКА МАТЕМАТИЧНОЇ МОДЕЛІ ТУРБО-КОДЕКА	21
2.1 Принципи турбо-кодування / декодування	21
2.2 Оптимізаційна задача пошуку найбільш правдоподібної послідовності ...	24
2.3 Алгоритм декодування Вітербі	28
2.4 Алгоритм декодування Вітербі з м'якими рішеннями	31
2.5 Висновки	42
3 ЕКСПЕРИМЕНТАЛЬНІ ДОСЛІДЖЕННЯ ЕФЕКТИВНОСТІ РОБОТИ.....	
ТУРБО-КОДЕКА.....	43
3.1 Вибір мови програмування	43
3.2 Опис роботи програми.....	45
3.3 Експерименти.....	46
3.4 Висновки	53
ВИСНОВКИ	54
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ	55
ДОДАТКИ	58
Додаток А (обов'язковий). Технічне завдання	59
Додаток Б (обов'язковий). Ілюстративна частина.....	62
Додаток В (обов'язковий). Лістинг програмного забезпечення.....	65
Додаток Г (обов'язковий). Протокол перевірки БКР	68

ВСТУП

Актуальність. Нині відбувається інтенсивний розвиток інформаційно-комунікаційних систем різного функціонального призначення (супутниковий і мобільний зв'язок, цифрове телебачення, радіосистеми) [1]. Для передавання даних вони використовують різні канали зв'язку (електричні, волоконно-оптичні, бездротові), у яких на сигнал діють завади різної фізичної природи, створюючи помилки у даних [2]. Для боротьби із завадами застосовується завадостійке кодування даних. Зі збільшенням трафіку та швидкостей передавання інформації зростає інтерес до ефективних технік завадостійкого кодування, із яких виділяються згорткові турбо-коди, рекомендовані до використання у сучасних стандартах зв'язку, оскільки дозволяють працювати на показниках, близьких до теоретичної межі [3].

Зважаючи на постійне зростання обсягів та швидкості передавання даних, суттєво збільшуються вимоги до цілісності та достовірності інформації, особливо в умовах низького відношення сигнал/шум. Тому розробка методів і засобів захисту інформації на основі завадостійкого кодування набуває особливої важливості і актуальності. Доцільно застосовувати алгоритми декодування на основі пошуку максимально правдоподібної послідовності бінарних символів, із яких виділяється алгоритм Вітербі з м'якими рішеннями та його двонаправлена версія [4]. Вони активно розвиваються, наприклад, можна навести наукові роботи закордонних та вітчизняних авторів, серед яких Ксі-Дж. Кся [5], Ф.Л. Моргос [6], В.Х.С. Лі [7], Ю.Ю. Іванов [8, 9] та інші. Тому, актуальною прикладною задачею є дослідження даних алгоритмів турбо-

декодування з метою аналізу ефективності їх роботи у системах передавання цифрових даних при різних умовах зовнішнього середовища.

Метою роботи є розширення функціональних можливостей програмного забезпечення, в якому реалізовано алгоритми турбо-декодування на основі пошуку максимально правдоподібної послідовності бітів, за рахунок розробки модуля для їх детального експериментального дослідження.

Для досягнення мети необхідно розв'язати такі *задачі*:

- проаналізувати роботу завадостійких кодів у засобах обміну даними;
- розробити математичний апарат алгоритмів турбо-декодування на основі пошуку максимально правдоподібної послідовності бінарних символів;
- розробити програмні засоби та дослідити розроблений турбо-кодек за допомогою імітаційного моделювання.

Об'єктом дослідження є процес обміну даними із забезпеченням необхідної вірогідності їх передавання.

Предметом дослідження є принципи, моделі, методи та інструментальні засоби турбо-декодування у системах передавання цифрових даних.

Методи дослідження. У роботі застосовано поняття обробки, збереження та передавання даних для аналізу турбо-декодування; імітаційне моделювання для перевірки достовірності отриманих теоретичних положень. Експерименти та обробка даних виконані у середовищі *MatLab*.

Науково-технічний результат роботи полягає у тому, що на основі теоретичних положень розроблене математичне, алгоритмічне та програмне забезпечення, яке може використовуватися для дослідження ефективності роботи алгоритмів турбо-декодування на основі пошуку максимально правдоподібної послідовності бітів.

У роботі отримані результати, які мають *практичну цінність*, а саме: наведено експериментальні криві ефективності роботи алгоритмів турбо-декодування на основі пошуку максимально правдоподібної послідовності бітів.

Апробація результатів та публікації. За результатами даної роботи опубліковано тези доповіді [10] на Всеукраїнській науково-технічній конференції підрозділів ВНТУ (м. Вінниця, 2025).

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ РОБОТИ

1.1 Основна термінологія

Передавання, отримання, обробка та зберігання інформації є основою функціонування будь-якої системи. В усіх цих процесах обов'язково здійснюється обмін даними між різними компонентами. У сучасних умовах широко використовуються системи цифрового зв'язку, де здійснюється передавання бінарних даних. Однак під час цього обміну неминуче виникають бітові помилки, спричинені шумами або перешкодами в каналі зв'язку, що призводить до спотворення переданих даних. Для зниження ймовірності виникнення помилок використовуються два основні підходи [11]:

1. Збільшення енергії сигналу перед передаванням. Це досить важко реалізувати, наприклад, якщо на розроблювану апаратуру систем зв'язку накладені різні обмеження: економічні, ергономічні і технічні (габарити передавача, його енергоспоживання тощо).

2. Застосування завадостійкого кодування (error correcting coding). Використовуються певні правила співставлення кожному конкретному повідомленню строго певної комбінації символів. Широке застосування завадостійкого кодування в сучасних цифрових системах зв'язку обумовлено тим, що воно є ефективним засобом приведення параметрів системи до бажаного компромісу між достовірністю, потужністю та швидкістю передавання даних.

Отже, використання завадостійкого кодування економічно доцільне. Такий вид кодування дозволяє зменшити ймовірність бітової помилки за

рахунок внесення надлишковості в передані повідомлення, тобто застосовується код, в якому використовуються не всі можливі комбінації, а лише деякі з них. Ця особливість на приймальній стороні використовується для боротьби з бінарними помилками. Відповідно такі коди називають надлишковими або коригуючими [2].

Процес введення надлишковості (перетворення інформаційних символів в кодове слово) називається кодуванням, а зворотній процес відновлення інформації з кодового слова декодуванням. Здатність завадостійких кодів визначати та виправляти помилки залежить від правил побудови цих кодів, їх параметрів та методів декодування.

Надлишковість коригуючого коду може бути абсолютною і відносною. Під абсолютною надлишковістю $r_{абс}$ розуміють кількість введених додаткових розрядів [2]:

$$r_{абс} = n - k, \quad (1.1)$$

де n – кількість кодових символів на виході кодера;

k – кількість інформаційних символів на його вході.

Відносною надлишковістю $r_{відн}$ коригуючого коду називають величину:

$$r_{відн} = \frac{r_{абс}}{n} = \frac{n - k}{n} = 1 - \frac{k}{n}. \quad (1.2)$$

Кодова відстань $d_{код}$ характеризує ступінь відмінності будь-яких двох кодових комбінацій. Вона виражається кількістю розрядів, в яких комбінації відрізняються одна від одної [1]:

$$d_{\text{код}}(X, Y) = W(X \oplus Y) = \sum_{i=1}^m |x_i - y_i|, \quad (1.3)$$

де W – вагова функція;

$X = (x_1, \dots, x_m)$, $Y = (y_1, \dots, y_m)$ – кодові слова (вектори).

Для завадостійкого кодування важливою є мінімальна кодова відстань $d_{\min}$ – найменша відстань Хеммінга між усіма можливими парами кодових комбінацій. Кратність гарантовано виявлених і виправлених помилок залежить від мінімальної кодової відстані $d_{\min}$. Під кратністю розуміється кількість пошкоджених завадами символів кодової комбінації.

Найбільш загальним показником для якісної оцінки методів декодування є енергетичний виграш від кодування (кодови резерв), який показує зниження енергії, необхідної для передавання даних, при певній середній ймовірності помилки на біт у разі використання заданих методів кодування / декодування, в порівнянні з випадком, коли кодування немає взагалі (рисунок 1.1) [11].

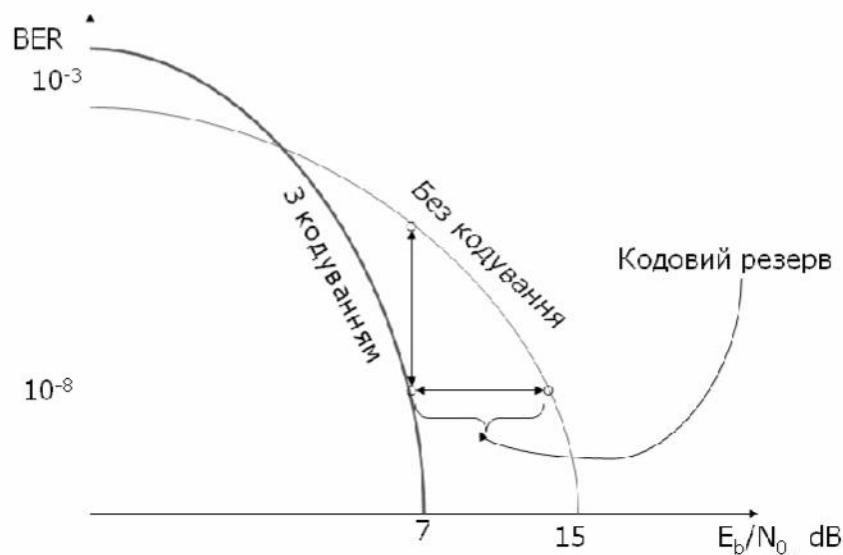


Рисунок 1.1 – Залежність частоти виникнення помилок BER від значення сигнал / шум E_b / N_0 (дБ)

Економічна цінність енергетичного виграшу надзвичайно актуальна і вимагає впровадження теоретичних досягнень з області завадостійкого кодування при створенні систем передавання цифрових даних різного функціонального призначення.

Існують різні найменування завадостійких кодів, але сучасна теорія інформації за принципами організації (побудови) простору кодових комбінацій розрізняє лише три основних напрямки: блокові коди, згорткові коди і каскадні коди, які можуть використовувати у своєму складі блокові та згорткові коди.

Поява методів каскадного кодування пов'язана з необхідністю значного спрощення методів декодування й одночасним підвищенням ефективності їх роботи в різних системах передавання даних. Розвиток каскадного кодування відбувається за двома напрямками. Перший із них пов'язаний з композицією блокових кодів шляхом послідовного з'єднання двох або більше кодерів, що забезпечує більшу швидкість передавання даних. Другий напрямок – згорткове кодування з використанням паралельного об'єднання двох і більше згорткових кодерів з системою перемішування символів. Використання згорткових компонентних кодів є найбільш перспективним, оскільки, порівняно з іншими методами декодування, дозволяє досягти рекордних показників BER на низьких відношеннях E_b / N_0 [12–14].

1.2 Методи завадостійкого кодування даних

Для опису методів кодування та декодування як блокових, так і згорткових кодів застосовують адекватний математичний апарат. Для лінійних кодів використовуються інструменти лінійної алгебри, тоді як формування

нелінійних кодів здійснюється за допомогою нелінійних процедур, які в деяких випадках дозволяють отримати коди з особливими властивостями.

У теорії завадостійкого кодування важливою є проблема складності реалізації методів кодування / декодування й, особливо, саме методів декодування. Тому певні класи блокових кодів (Хеммінга, Боуза-Чоудхурі-Хоквінгема, Ріда-Соломона, Файра) були розроблені разом з методами декодування, пов'язаними зі структурними властивостями цих кодів. І, навпаки, розробка нових методів декодування згорткових кодів (алгоритми Вітербі, Бала-Кока-Джелінека-Равіва, послідовне та порогове декодування) ініціювала пошуки відповідних згорткових кодів [11].

Формування блокових кодів передбачає розбиття переданих цифрових послідовностей на окремі блоки, які подаються на вхід кодера. Кожному такому блоку на виході кодера відповідає блок кодових символів, робота кодера визначається алгоритмом кодування. Блокові коди історично були запропоновані й вивчені на початку розвитку теорії кодування. Блоковий код представляє собою код без пам'яті. Кодер блокового коду відображає блок довжиною із k інформаційних символів, який подається на вхід, у кодову послідовність з n вихідних символів. Термін “без пам'яті” вказує на те, що кожен блок з n символів залежить тільки від відповідного блоку з k символів і не залежить від інших блоків [11].

Формування неперервних кодів здійснюється неперервно в часі, без поділу на блоки, що й визначає назву цього класу кодів. У даному класі кодів слід виділити згорткові коди, які за характеристиками перевершують блокові коди. Вони мають високу коригуючу здатність як для випадкових, так і для пакетних помилок. Кодер згорткового коду є пристроєм з пам'яттю.

Він представляє собою кінцевий автомат, який може бути заданий у вигляді функції [11]:

$$W = (V, Q, F, F', \lambda), \quad (1.4)$$

де V – вхідний алфавіт, з якого формується вхідна послідовність;

Q – множина станів;

F – множина початкових станів ($F \in Q$);

F' – множина кінцевих станів ($F' \in Q$);

λ – функція переходів між станами.

Автомат починає роботу в стані $f_0 \in F$, зчитуючи по одному символу вхідної послідовності. Зчитаний символ переводить автомат у новий стан відповідно до функції переходів. Прочитавши вхідну послідовність, автомат буде в стані $f_n' \in F'$. Фактично кодер згорткового коду можна розглядати як лінійний регістр зсуву з пам'яттю m , складність якого не залежить від довжини коду, що є значною перевагою. На його вхід (аналогічно до блокового) надходять послідовності з k вхідних інформаційних символів, а на виході з'являються послідовності з n кодових символів. Але при цьому кожна послідовність з n кодових символів залежить від поточної вхідної послідовності і від v попередніх вхідних символів. Отже, кодер повинен містити пристрій пам'яті на m вхідних символів, а параметр $K = m + 1$ називають довжиною кодового обмеження.

Якщо прийняти, що d_k – вхідний бінарний символ і a_k – бінарний символ на позиції k в регістрі зсуву (стан елемента), то для рекурсивного систематичного згорткового кодера ($K = 3$, $R = 0,5$) з рисунку 1.2 можна отримати значення дібітів [15]:

$$\{d_{k,1}, d_{k,2}\} = \{d_k, d_k \oplus a_{k-1} \oplus a_{k-2}\}. \quad (1.5)$$

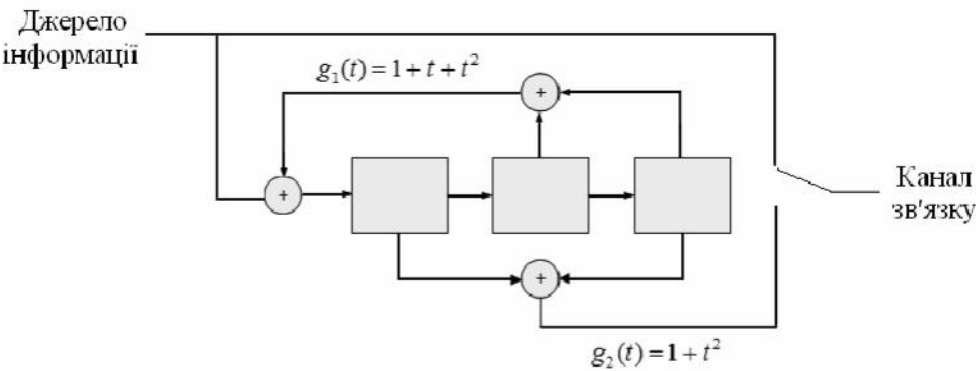


Рисунок 1.2 – Структура згорткового кодера

Згорткові коди можна описати решітчастою діаграмою (трелліс), що дозволяє реалізувати ймовірнісне декодування. Отже, роботу кодера зручно розглядати на основі теорії графів, побудувавши діаграми станів, які визначаються за допомогою значень комірок пам'яті m . Нехай у деякий момент часу стан кодера з генератором $g(t_i) = (7, 5)_8$ та пам'яттю $m = 2$ буде 00. Всі можливі переходи зі станів кодера наведено в таблиці 1.1.

Таблиця 1.1 – Кодування бітової послідовності

$d_k = d_{k,1}$	d_k	a_{k-1}	a_{k-2}	a_k	a_{k-1}	$d_{k,1}$	$d_{k,2}$
0	0	0	0	0	0	0	0
	1	1	0	1	1	0	1
	1	0	1	1	0	0	0
	0	1	1	0	1	0	1
1	1	0	0	1	0	1	1
	0	1	0	0	1	1	0
	0	0	1	0	0	1	1
	1	1	1	1	1	1	0

На їх основі можна побудувати орієнтований граф або повну діаграму станів кодера ($g(t_i) = (7,5)_8$, $K = 3$, $R = 0,5$), де у кружечках визначаються стани кодера S , а вхідний символ d_k і дібіт $d_1 d_2$ пишуть через косу лінію (рисунок 1.3).

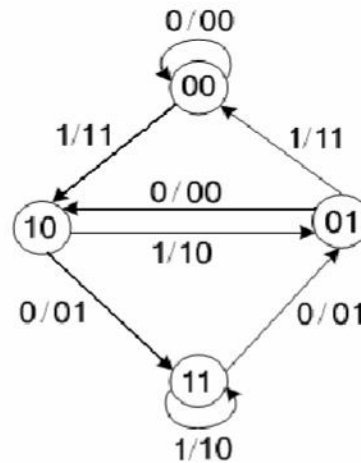


Рисунок 1.3 – Повна діаграма станів кодера

Використовуючи діаграму станів, можна побудувати часову діаграму переходів для послідовності бітів. Подібну структуру у формі трелліс-діаграми зображено на рисунку 1.4.

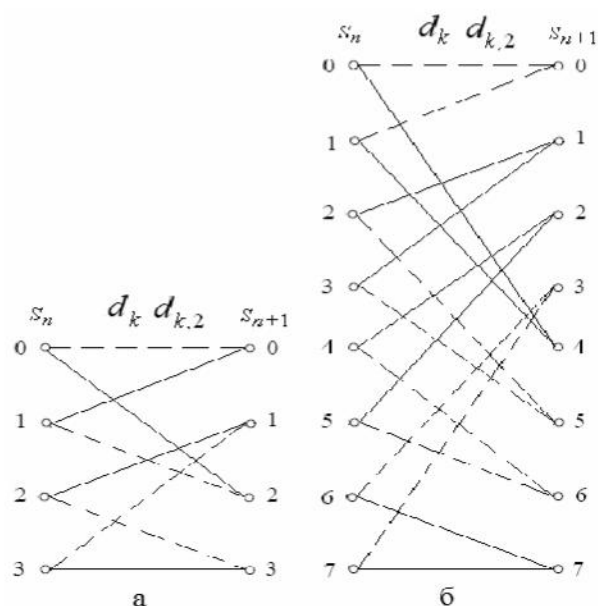


Рисунок 1.4 – Діаграма згорткового коду: (а) $K = 3$, $R = 0,5$; (б) $K = 4$, $R = 0,5$

Відмітні переваги коригуючих кодів (як блокових, так і згорткових) спонукали пошуки нових підходів до реалізації шляхів підвищення завадостійкості й ефективності систем цифрового зв'язку. Результатом цих пошуків стала каскадна конкатенація блокових або згорткових кодів. Такий код представляє собою високоефективний каскадний блоковий систематичний код з високою коригуючою здатністю та спеціальними властивостями компонентних кодів. На практиці для турбо-подібних кодів існують певні реалізаційні особливості, про які слід згадати. Типова крива частоти виникнення помилок (Bit Error Rate, BER) має три основні області: спочатку відбувається поступове зменшення BER – область неузгодженості, потім різке зменшення – ”скеляста область”, а потім зустрічається область “плато насичення”, в якій нахил кривої різко зменшується (рисунок 1.5). При цьому, для досягнення малої величини $BER = 10^{-9}$ – 10^{-11} необхідні значні витрати енергетичних ресурсів [10, 14, 16].

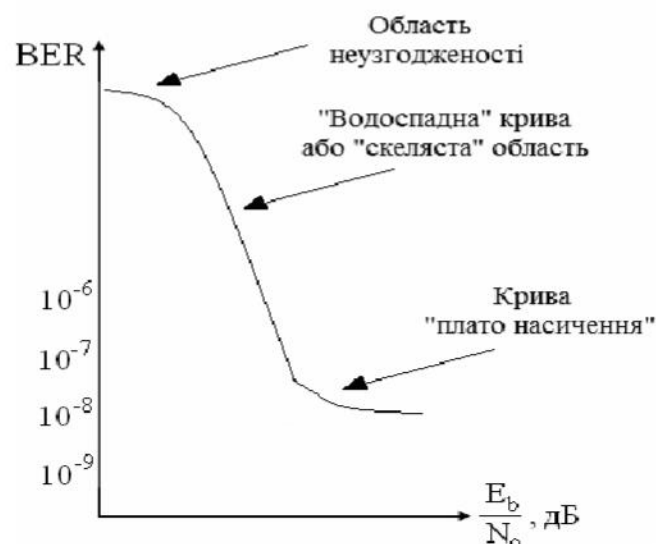


Рисунок 1.5 – Узагальнена характеристика BER для турбо-подібних кодів

При високих імовірностях виникнення помилки $BER = 10^{-5} - 10^{-7}$ (мале відношення сигнал/шум, від 0 до 3 дБ) краще показує себе згортковий турбокод. Його недоліками є відносно висока складність практичної реалізації, а також наявність явної області насичення ймовірності помилки. Блоковий турбокод має значно більшу кодову відстань, а область насичення ймовірності помилки для нього буде набагато нижчою. Саме тому даний код є оптимальним вибором при низьких $BER = 10^{-8} - 10^{-12}$ (велике відношення сигнал/шум), оскільки він може забезпечувати дуже малі ймовірності помилки декодування при високій швидкості передавання даних для малих та середніх пакетів [16].

Основним напрямком в розвитку теорії кодування є захист цілісності інформації на основі паралельних каскадних конструкцій згорткових турбокодів. Головний принцип турбо-кодування – використання декількох компонентних кодерів, що працюють паралельно. При цьому інформаційний блок кодується декілька разів згідно кількості застосованих у системі кодів, причому, другий і наступні кодери здійснюють процедуру кодування лише після попереднього перемішування символів (рисунок 1.6) [9, 17].

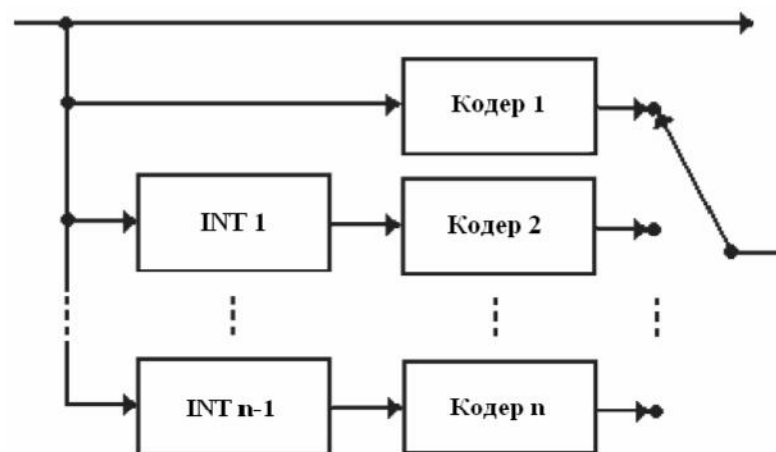


Рисунок 1.6 – Кодова структура згорткового турбо-коду

Декодування турбо-кодів здійснюється ітеративним декодером, складові частини якого виконують імовірнісне декодування з використанням м'яких рішень. Такий підхід дозволяє отримати дійсне число, яке є мірою ймовірності помилки при декодуванні біта (міру надійності жорсткого рішення).

Існує дві основні групи методів декодування турбо-кодів з використанням "м'якого" прийняття рішення. Перша з них мінімізує ймовірність помилки біта D_k як максимум апостеріорної ймовірності. Інша мінімізує ймовірність помилки всієї інформаційної послідовності X . При цьому в обох випадках оцінюється правдоподібність та максимізується кореляційна метрика [18–20]. У даній роботі розглядається група, яка базується на алгоритмі декодування Вітербі з м'яким виходом (пошук максимально правдоподібної послідовності).

Загалом майже всі потужні компанії, які є світовими експертами в області комунікацій (Comtech, Turbo Concept, Xilinx), використовують саме згорткові (ЗТК) та блокові турбо-коди (БТК) та код з низькою щільністю перевірок на парність (КНЦПП), що показано у таблиці 1.2.

Таблиця 1.2 – Використання кодів у сучасних стандартах зв'язку

Компанія	Тип коду	Стандарт зв'язку
France Telecom	ЗТК, БТК	W-CDMA, CDMA2000, IEEE 802.11 та 802.16
Turbo Concept	ЗТК, КНЦПП, БТК	DVB-S2, DVB-RCS, IEEE 802.16
Comtech	ЗТК, КНЦПП, БТК	W-CDMA, CDMA2000, DVB-S2, IEEE 802.16
Xilinx	ЗТК, КНЦПП, БТК	W-CDMA, DVB-S2, IEEE 802.16

Турбо-подібні коди застосовуються в цифровому зв'язку (цифрове телебачення, супутниковий та мобільний зв'язок, програмовані радіосистеми тощо), ефективно вирішуючи задачу досягнення максимальної ефективності передавання даних у зашумленому каналі при обмеженій смузі частот. Важливою особливістю таких кодів є те, що вони дозволяють при розумних обчислювальних витратах наблизитися до теоретичної межі завадостійкості [3].

1.3 Дослідження об'єкта автоматизації

У даній роботі розглядається автоматизована інформаційна система, яка виконує обробку інформації та обмін даними. Основна задача системи передавання даних – обмін даними із заданими вірогідністю та швидкістю передавання, причому ці вимоги перебувають у протиріччі, тобто підвищення швидкості передавання призводить до зниження завадостійкості та вірогідності передавання даних [2].

Автоматизована система є організаційно-технічною системою, що дозволяє отримати рішення на основі автоматизації інформаційних процесів у різних сферах діяльності (управління, проектування, виробництво тощо) або їх поєднаннях. Проектування її структурно-алгоритмічної частини містить розробку складу та структури інформаційних потоків та алгоритмів їх обробки, включаючи загальний алгоритм функціонування системи. Технічні засоби для автоматизації управління процесами виконують такі функції: збирання, оброблення та перетворення інформації про стан процесу; передавання інформації каналами зв'язку; формування команд управління; використання та подання командної інформації та зв'язок з оператором [21].

Отже, необхідно дослідити процес обробки даних, організацію роботи з ними, розглянути об'єкт автоматизації, наприклад, кодек, описати дані, які подано на його вхід та отриманий результат. Інформація від джерела до адресата передається відповідно до структурної схеми, наведеної на рисунку 1.7. На ньому $\bar{U}$ – початкові дані, які надаються кодеру; $\bar{V}$ – дані після кодування; $\bar{X}$ – дані після модуляції, які надходять до каналу із шумом; $\bar{Y}$ – дані на виході каналу з шумом; $\bar{r}$ – дані після демодуляції, які надходять до декодера; $\tilde{U}$ – отримана інформація після проходження вузлів системи [20].



Рисунок 1.7 – Узагальнена структура для роботи з інформацією

Відправник інформаційного повідомлення є джерелом інформації і передає дані інформаційним каналом, в якому можуть бути присутні шумові ефекти, адресату. Прийом повідомлення є основною ланкою інформаційного процесу і представляє собою відношення: предмет (ім'я, носій інформації) – приймач інформації – контекст – інтерпретація повідомлення.

Кодер системи отримує інформаційні символи від джерела і додає до них надлишкові символи таким чином, щоб могла бути виправлена значна частина помилок, що виникають під час передавання даних каналом з шумом.

Закодовані символи надходять на модулятор, який перетворює їх в аналоговий сигнал. Демодулятор навпаки перетворює прийнятий сигнал в послідовність чисел, які представляють оцінку переданих даних (метрики).

Вони надходять у декодер, який виправляє помилки, приймає рішення про символ, використовуючи внесену кодером надлишковість, та передає його до приймача.

Процес двійкової фазової модуляції описується формулою (1.6), а демодуляції – рівнянням (1.7), де E_c – енергія кодованого символу:

$$\overline{X}_i = (-1)^{\overline{V}_i+1} \cdot \sqrt{E_c}, \quad (1.6)$$

$$\overline{r}_i = \begin{cases} -\sqrt{E_c} \Rightarrow \overline{V}_i = 0; \\ \sqrt{E_c} \Rightarrow \overline{V}_i = 1. \end{cases} \quad (1.7)$$

Дійсні величини шуму ξ з каналу зв'язку додаються до символів з модулятора, формуючи зашумлений сигнал, у формі:

$$Y_{\xi_i} = \overline{X}_i + \xi. \quad (1.8)$$

Якщо на значення величини впливає багато випадкових факторів, вона в підсумку буде розподілена нормально. Тому для моделювання застосовують канал зв'язку з адитивним білим гаусівським шумом.

Застосування завадостійких кодів в автоматизованих системах дозволяє підвищити надійність каналів зв'язку, особливо в умовах обмежених ресурсів. У реальних системах дані коди є важливою частиною автоматичних механізмів коригування помилок та забезпечення стабільного зв'язку.

1.4 Висновки

У розділі представлено основні поняття теорії передавання даних та систему цифрового зв'язку. Аналіз показує, що сучасні тенденції підтверджують перевагу згорткових турбо-кодів у каналах із високим рівнем шуму, в яких вони використовуються з ітеративними методами декодування. У наступному розділі буде розглянуто математичну модель турбо-кодека на основі ітеративного пошуку максимуму правдоподібності послідовності бітів.

2 РОЗРОБКА МАТЕМАТИЧНОЇ МОДЕЛІ ТУРБО-КОДЕКА

2.1 Принципи турбо-кодування / декодування

У класичній схемі каскаду турбо-кодів використовуються два однакових рекурсивних систематичних згорткових кодера, що дозволяє отримати достатньо високу вагу кодових слів. Низька вхідна вага кодових слів є основним фактором виникнення помилок. Турбо-кодер, зображений на рисунку 2.1, складається з двох паралельно з'єднаних згорткових кодерів. Відмінність між ними полягає в тому, що перший містить систематичний вихід, через який до каналу передавання даних передається інформаційна послідовність, що гарантує систематичне представлення кодової послідовності.

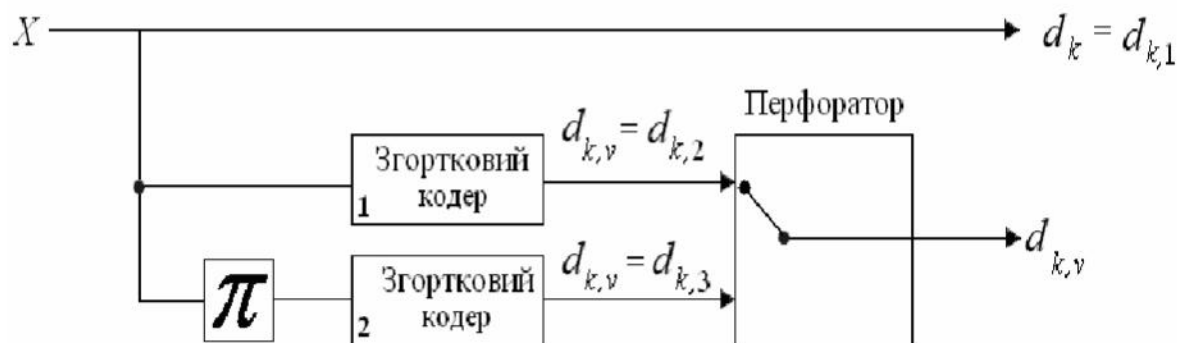


Рисунок 2.1 – Структура турбо-кодера

З виходу всього турбо-кодера без процедури перфорації на модулятор спочатку поступає символ з систематичного виходу верхнього кодера $d_{k,1}$, а потім – два перевіірочних символи: з першого кодера $d_{k,2}$ та другого $d_{k,3}$. Отже, двійкові послідовності на виході кодерів являють собою перевіірочні символи, які разом з інформаційними бітами складають єдине кодове слово. Завдяки

використанню рекурсивних кодерів в кодовому блоці явно виділяється систематична і перевірна частини [9, 17].

Можна вважати, що до каналу зв'язку передаються два кодові блоки: перший блок, який складається з інформаційної частини $d_k = d_{k,1}$ і перевірної частини $d_{k,2}$ кодера 1, і другий кодовий блок, що складається з перемішаної інформаційної частини $\pi(d_k)$ і перевірної частини $d_{k,3}$ кодера 2.

Аналітичне моделювання каналів передавання інформації у теорії завадостійкого кодування зазвичай передбачає гаусівський розподіл завад, що означає статистичну незалежність потоку помилок. Проте для реальних каналів зв'язку таке припущення не завжди виправдане, оскільки в цих системах помилки часто мають тенденцію до групування у пакети помилок. Дане явище створює труднощі при декодуванні завадостійких кодів [3].

Для подолання цього ефекту, тобто декореляції інформаційної послідовності в часі, отримання квазівипадкових послідовностей, які є відомими отримувачу, а також для збільшення мінімальної відстані, застосовується принцип перемішування.

При декодуванні згорткового турбо-коду (рисунк 2.2) із загального потоку даних виділяють два зашумлені кодові блоки $(x_{\xi_k}, x_{\xi_{k,2}})$ та $(x_{\xi_k}, x_{\xi_{k,3}})$, причому їх інформаційні частини через систематичне кодування з урахуванням перемішування ідентичні. Ця обставина дозволяє використовувати два декодери, кожен з яких проводить декодування свого кодового блоку.

Принцип подачі вихідних даних назад на вхід аналогічний турбонаддуву двигуна з основним робочим елементом – турбокомпресором, тому турбо-код має саме таку назву [14].

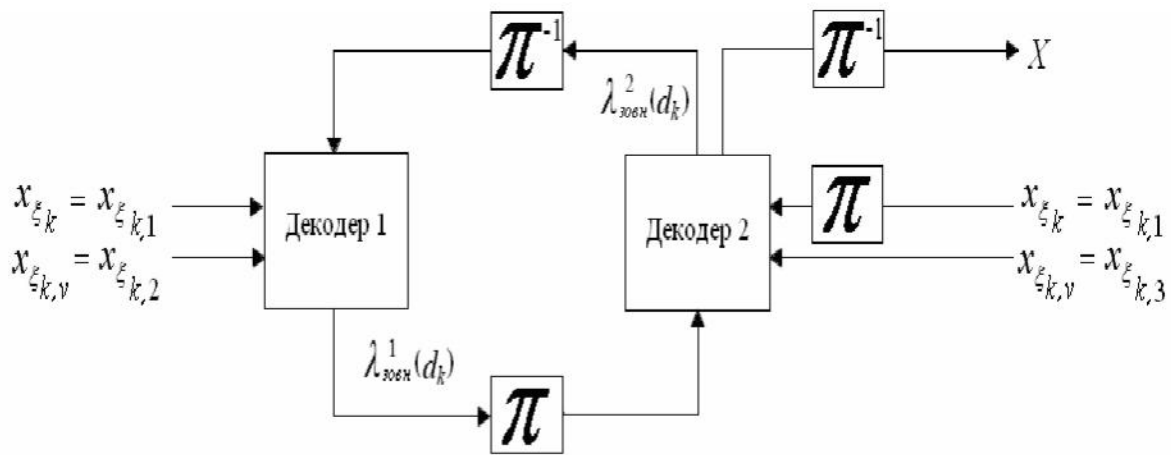


Рисунок 2.2 – Структурна схема турбо-декодера

Це означає, що декодовану інформацію першого (другого) декодера $\lambda_{зовн.}(d_k)$ (зовнішня інформація) з урахуванням перемішування π можна використовувати як апіорну інформацію $\lambda_{ап.}(d_k)$ для другого (першого) декодера з метою уточнення результату декодування, тобто інформація обертається петлею до того моменту, поки м'які рішення не сходяться на стабільному наборі значень. Крім того, для уточнення апостеріорного $\lambda(d_k)$ використовується канална інформація $\lambda_{кан.}$. Подібну операцію можна проводити багаторазово (рисунок 2.3). Така ітеративна взаємодія між декодерами дозволяє значно покращити точність відновлення переданих даних, знижуючи ймовірність помилок навіть за умов зашумлених каналів зв'язку [10, 22, 23].

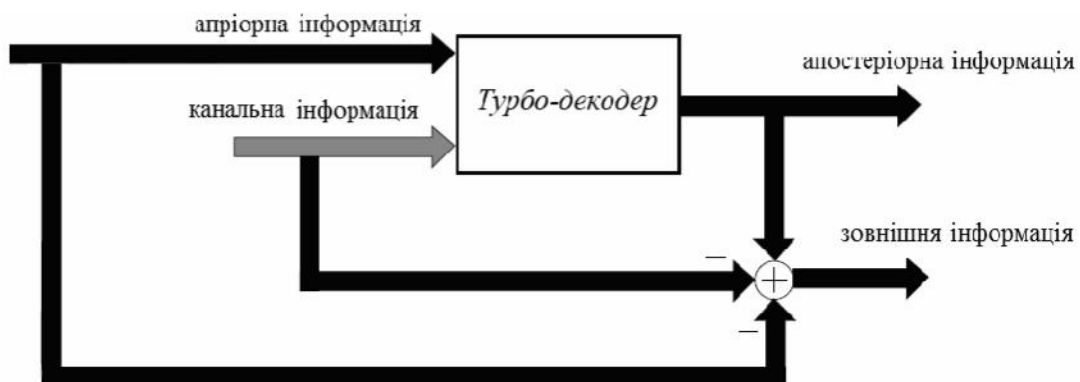


Рисунок 2.3 – Принцип турбо-декодування

Зупинка ітеративного процесу турбо-декодування відбувається або після виконання заданої кількості ітерацій q , або після того, як значення результату декодування за певним критерієм зупинки досягне встановленого порогу (критерій на основі зміни м'яких та жорстких рішень; з розрахунком контрольної суми тощо). Для моделювання використовують ідеальний теоретичний критерій "магічного джина" [9].

2.2 Оптимізаційна задача пошуку найбільш правдоподібної послідовності

У нашому випадку розглядається екстремальна задача пошуку найбільш правдоподібного шляху на гратці згорткового коду під час декодування. Ця задача є класичним прикладом оптимізаційної задачі, для аналізу якої можна застосувати динамічне програмування. Для того, щоб нашу задачу можна було описати моделлю динамічного програмування, необхідно виконання низки умов [1, 11, 15]:

- 1) задача може інтерпретуватись як n -кроковий процес управління, а показник ефективності процесу може бути представлений в адитивній формі, тобто як сума показників ефективності на кожному кроці;
- 2) структура задачі інваріантна відносно кількості кроків n , тобто повинна бути визначена для будь-якого n і не залежати від нього;
- 3) на кожному кроці стан системи визначається кінцевою кількістю параметрів стану та управляється кінцевою кількістю змінних;
- 4) вибір рішення на k -ому кроці не впливає на попередні кроки, а стан на початку цього кроку є функцією лише попереднього стану і вибраного на ньому рішення.

Динамічне програмування – метод оптимізації, пристосований до операцій, в яких процес прийняття рішень може бути розділений на окремі етапи (кроки), причому кожний крок визначається лише поточним станом і не залежить від попередніх подій. Метод динамічного програмування широко застосовується для оптимізації дискретних систем (рисунок 2.4). До типових задач динамічного програмування відносяться задачі про розрахунок траєкторії літака, рюкзак, інвестиції, розклад поїздів, завантаження торгового судна.

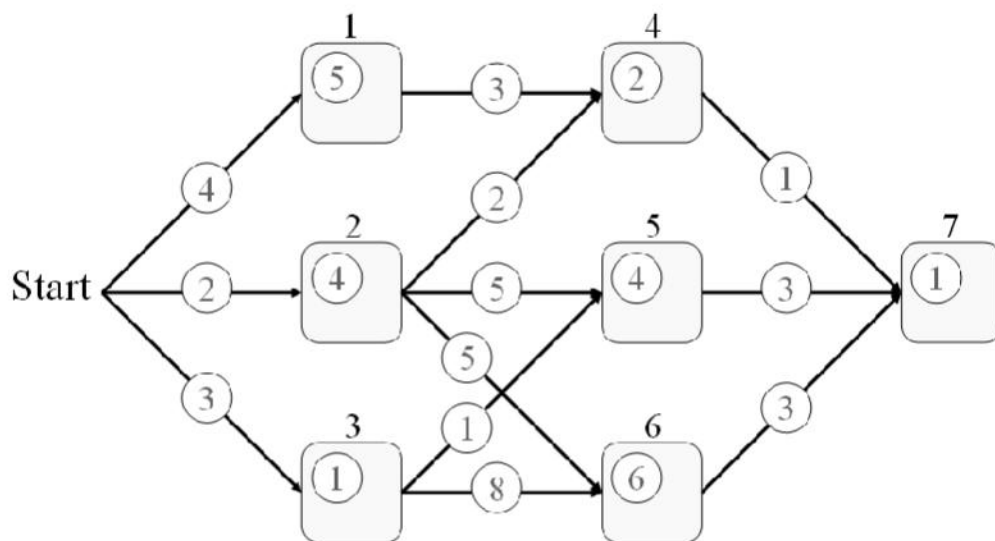


Рисунок 2.4 – Приклад роботи динамічного програмування

Фактично динамічне програмування представляє собою математичний апарат, який дозволяє здійснювати оптимальне планування багатокрокових керованих процесів. Початком етапу керованого процесу вважається момент прийняття рішення. Плануючи поетапний процес, виходять з інтересів усього процесу в цілому, тобто при прийнятті рішення на окремому етапі завжди необхідно мати на увазі кінцеву мету [1].

Динамічне програмування, використовуючи поетапне планування, дозволяє не тільки спростити рішення задачі, але і вирішити ті з них, до яких не

можна застосувати методи математичного аналізу. Поетапність аж ніяк не припускає ізольованості в оптимізації етапів. Навпаки, управління на кожному кроці вибирається з урахуванням усіх його наслідків. Ідея поступової, покрокової оптимізації становить суть методу динамічного програмування. Оптимізація одного кроку, як правило, простіша оптимізації усього процесу в цілому. Краще багато разів вирішувати просту задачу, ніж один раз – складну.

Основною перевагою такого підходу можна вважати те, що розробники займаються одновимірними оптимізаційними підзадачами замість багатовимірної основної задачі, рішення якої збирається “знизу – вгору”. Метод вдосконалює виконання задач, що вирішуються за допомогою перебору варіантів або рекурсій. Істотним недоліком даного методу є великий об’єм обчислень і необхідної пам’яті для зберігання проміжних результатів при рішенні задач на ЕОМ, причому, вимоги до пам’яті сильно зростають при збільшенні розмірності задачі. Наприклад, якщо для дослідження кожної змінної одновимірної задачі потрібно 10 кроків, то в двовимірній задачі їх кількість збільшується до 100, в тривимірній – до 1000 і т.д. Крім того, на відміну від лінійного програмування, в якому симплексний метод є універсальним, в динамічному програмуванні такого методу не існує. Кожне завдання має свої труднощі, і в кожному випадку необхідно знайти найбільш підходящу методику вирішення. Динамічне програмування ґрунтується на принципах Беллмана [11].

Принцип оптимальності. Оптимальний розв’язок на кожному кроці визначається станом системи на початку цього кроку. Яким би не був стан системи в результаті певної кількості кроків, на найближчому кроці потрібно вибирати стан на трелліс-гратці таким чином, щоб він в сукупності з

оптимальним вибором на всіх наступних кроках призводить до оптимального виграшу на всіх кроках, що залишилися, включаючи даний.

Основна вимога роботи принципу: відсутність зворотного зв'язку, тобто управління процесом на даному кроці не повинно здійснювати вплив на попередні кроки.

Принцип занурення. Форма завдання, яке розв'язується методом динамічного програмування, не змінюється при зміні кількості кроків n . У цьому сенсі всякий конкретний процес із заданим числом кроків виявляється як би зануреним в сімейство подібних йому процесів і може розглядатися з позиції більш широкого класу задач.

Реалізація названих принципів дає гарантію того, що рішення, прийняте на черговому кроці, виявиться найкращим щодо всього процесу в цілому. Послідовність покрокових рішень призводить до вирішення вихідної задачі.

Представимо математичне формулювання принципу оптимальності. Для простоти будемо вважати, що задані початковий s_0 і кінцевий s_N стани системи. Позначимо через $z_1(s_0, u_1)$ значення цільової функції на першому етапі при початковому стані системи s_0 і при управлінні u_1 , через $z_2(s_1, u_2)$ – відповідне значення цільової функції на другому етапі, $z_i(s_{i-1}, u_i)$ – на i -му етапі, $z_N(s_{N-1}, u_N)$ – на N -му етапі. Потрібно знайти оптимальне управління u^* . Для вирішення цієї задачі занурюємо її в сімейство подібних. Позначимо через $F_1(s_{N-1}), F_2(s_{N-2}), \dots, F_k(s_{N-k}), F_N(s_0)$ відповідно умовно-оптимальні значення цільової функції z на останньому етапі, двох останніх, на k останніх, на всіх N етапах. Починаємо з останнього етапу. Нехай s_{N-1} – можливий стан системи на початок N -го етапу. Тоді визначаємо, що:

$$F_1(s_{N-1}) = z_N(s_{N-1}, u_N), \quad (2.1)$$

$$F_2(s_{N-2}) = z_{N-1}(s_{N-2}, u_{N-1}) + F_1(s_{N-1}), \quad (2.2)$$

$$F_3(s_{N-3}) = z_{N-2}(s_{N-3}, u_{N-2}) + F_2(s_{N-2}). \quad (2.3)$$

$$F_k(s_{N-k}) = z_{N-k+1}(s_{N-k}, u_{N-k+1}) + F_{k-1}(s_{N-k+1}), \quad (2.4)$$

$$F_N(s_0) = z_1(s_0, u_1) + F_{N-1}(s_1). \quad (2.5)$$

Вираз (2.5) являє собою математичний запис принципу оптимальності. Формула (2.4) – загальна форма запису умовно-оптимального значення цільової функції, якщо залишилося k станів. Вирази (2.1)-(2.5) називаються функціональними рівняннями Беллмана. Проглядається їх рекурентний характер, тобто для знаходження оптимального управління на N кроках потрібно знати умовно-оптимальне управління на попередніх $N-1$ етапах і т.д. Тому функціональні рівняння часто називають рекурентними (зворотними) співвідношеннями Беллмана.

2.3 Алгоритм декодування Вітербі

Декодер Вітербі у разі безпомилкового приймання всієї послідовності дібітів має повну інформацію про цю послідовність, а також про структуру кодера (тобто його діаграму станів) та початковий стан. Із цією інформацією він повинен відновити початкову послідовність бітів, використовуючи відповідні таблиці і метрики помилок. Для переходу, що відповідає прийнятому дібіту (тобто перехід правильний), метрика помилок приймається рівною нулю, а для інших переходів вона розраховується за кількістю відмінних бітів в прийнятому дібіті та тому дібіті, який відповідає розглянутому переходу [20].

Наприклад, якщо прийнятий дібіт 00, а дібіт, що відповідає переходу, рівний 11, то метрика помилок (метрика Хеммінга) для даного переходу дорівнює 2. Отже, будується трелліс-гратка, на якій прокладається шлях, що має найкоротшу метрику накопичених помилок за всі такти t кодера, тобто для пошуку екстремуму можна задати цільову функцію z наступним чином [10]:

$$z = error_metric(t_1, t_2, \dots, t_n) \rightarrow \min. \quad (2.6)$$

Для цього зручно побудувати таблицю, в якій відображати всі метрики помилок для переходів із одного стану в інший за певний момент часу. Ті шляхи, які мають найменшу метрику помилок, називають “вижившими” [18].

Нехай декодер отримав послідовність виду

$$00\ 11\ 10\ 00\ 01\ 10\ 01\ 11\ 11\ 10\ 00\ 10\ 11\ 00\ 11\ 10\ 11,$$

тоді його робота починається з лівого верхнього вузла трелліс-діаграми (як і робота кодера). У цій точці декодер аналізує ситуацію, коли кодер міг знаходитися в стані 00 і на його вході з'явився 0. Таким чином, в канал зв'язку кодер міг відправити значення 00 (рисунок 2.5).

Порівнюючи ці показники з показниками, які реально прийняв приймач (дані з каналу зв'язку відповідають значенню дібіта 00), декодер встановлює вагу цього шляху, використовуючи порозрядне складання за $\text{mod } 2$ даних із каналу зв'язку і даних вироблених декодером для даного ребра, тобто

$$00 \oplus 00 = 00.$$

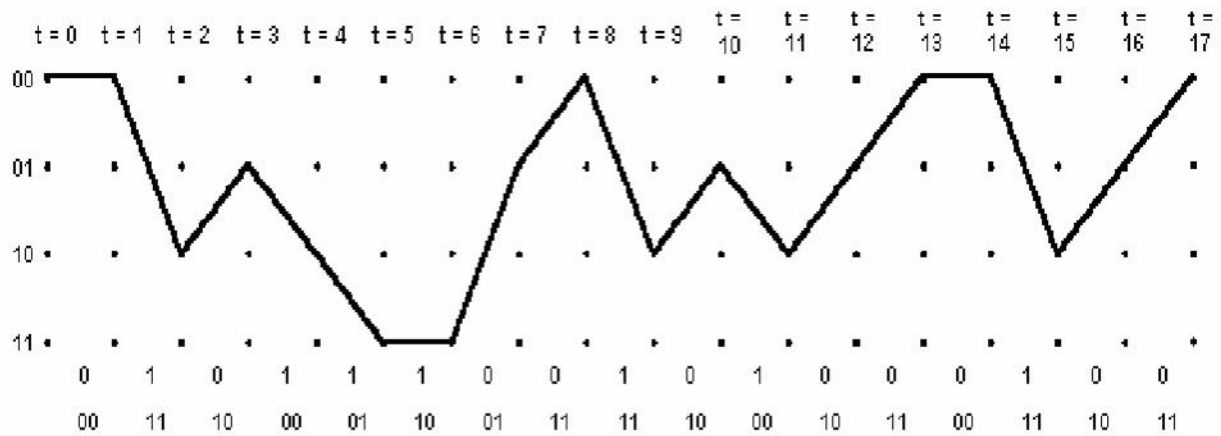


Рисунок 2.5 – Декодування за алгоритмом Вітербі

Відповідно вага цього шляху дорівнює 0, тобто зафіксований повний збіг інформації, прийнятої із каналу зв'язку і інформації, яка з'являється на виході декодера в стані 00. Якщо ж декодер перевіряє варіант дії кодера, коли стан його елементів пам'яті міг відповідати, наприклад стану 10, то у цій ситуації кодер повинен був відправити до каналу зв'язку пару бітів 11, тоді вага цього ребра дорівнює 2, тобто ймовірна помилка:

$$00 \oplus 11 = 11.$$

Якщо на черговому кроці з'являється пара напрямів, вага яких виявляється рівною 0, то в цьому випадку перевага віддається тому шляху, який продовжує попередній шлях мінімальної ваги. На першому кроці подібний аналіз виконувався всього для двох ребер графа, на другому кроці вже чотири ребра, а потім всі 2^m ребер, де m – кількість комірок пам'яті згорткового кодера. Фактично, як вже було сказано, алгоритм декодування Вітербі базується на ідеях динамічного програмування Беллмана.

2.4 Алгоритм декодування Вітербі з м'якими рішеннями

Декодер з “м'яким” прийняттям рішення, крім бінарного значення 1 або 0, також приймає довірчу величину, яка залежить від заданого біта. Надійність м'яких рішень оцінюється їх абсолютним значенням: чим воно більше і далі від нуля, тим вища надійність рішення щодо бінарного символу d_k . Знак м'якого рішення λ вказує на те, яке рішення було прийнято на поточному етапі. Якщо демодулятор визначився, то біту присвоюється значення 1, і ступінь упевненості в ньому буде високою. Якщо ж визначеність менша, то й довірна величина буде низькою [9]:

$$\tilde{U}_k = \text{sign}(\lambda(d_k)) = \frac{\lambda(d_k)}{|\lambda(d_k)|} = \begin{cases} 1, & \lambda(d_k) > 0; \\ 0, & \lambda(d_k) \leq 0. \end{cases} \quad (2.7)$$

Алгоритм Вітербі з м'якими рішеннями розглядає мінімізацію ймовірності помилки послідовності X , тобто являє собою декодування за максимумом правдоподібності для послідовності бітів. Для роботи даного алгоритму декодування потрібно використати метрики ребер на ґратці коду та блок операцій додати-порівняти-вибрати [9, 18].

Спочатку обчислюється метрика шляху $M_k(s)$. Для переходу, що відповідає прийнятій послідовності (тобто перехід правильний), м'яке рішення для метрики шляху буде більшим. Будується трелліс, на якій прокладається шлях, що має найбільше значення метрик шляху $M_k(s)$ за всі такти кодера. Шляхи, які мають більшу метрику шляху $M_k(s)$, є “вижившими”. На основі аналізу зростання найбільш правдоподібної послідовності визначають оптимальний шлях. Після знаходження шляху з максимальною кінцевою

метрикою можна застосувати класичний підхід Вітербі або апроксимаційну формулу Хагенауера для пошуку логарифмічного відношення функцій правдоподібності кожного символу на трелліс [17].

На основі праць [8, 9, 18, 20, 24] представимо математичну модель даного алгоритму декодування. Для цього використовується поняття марківського ланцюга з множиною станів $\{s_0, \dots, s_n\}$ (стани ґратки) та враховується, що отримано зашумлену послідовність бітів $\{x_{\xi_0}^n\}$. Проводиться пошук послідовності станів s_i (шляху) на трелліс і максимізується апостеріорна ймовірність $p(\{s_0, \dots, s_n\} | x_{\xi_0}^n)$, яку, використавши умовну ймовірність, можна записати як:

$$p(\{s_0, \dots, s_n\} | x_{\xi_0}^n) = \frac{p(x_{\xi_0}^n, \{s_0, \dots, s_n\})}{p(x_{\xi_0}^n)} = \frac{p(x_{\xi_0}^n | \{s_0, \dots, s_n\}) \cdot p(\{s_0, \dots, s_n\})}{p(x_{\xi_0}^n)}. \quad (2.8)$$

Отже, на основі аналізу формули (2.8) необхідно максимізувати взаємну ймовірність у чисельнику [18]:

$$\Lambda = p(x_{\xi_0}^n, \{s_0, \dots, s_n\}) = p(x_{\xi_0}^n | \{s_0, \dots, s_n\}) \cdot p(\{s_0, \dots, s_n\}) \rightarrow \max. \quad (2.9)$$

Дану задачу оптимізації можна розв'язати, врахувавши, що $\{s_0, \dots, s_n\}$ представляє собою марковську послідовність, а отримана послідовність $\{x_{\xi_0}^n\}$ не залежить від вибору шляху на решітці, тобто відбувається ряд незалежних подій (отримання символів), що за теоремою множення ймовірностей можна записати наступним чином:

$$\begin{aligned}\Lambda &= p(x_{\xi_0}^{n-1}, \{s_0, \dots, s_{n-1}\}) \cdot p(x_{\xi_n}, s_{n-1}, s_n) = \\ &= p(x_{\xi_0}^{n-1}, \{s_0, \dots, s_{n-1}\}) \cdot p(x_{\xi_n} | d_k) \cdot p(d_k).\end{aligned}\quad (2.10)$$

Розглянемо вираз $p(x_{\xi_n} | d_k) \cdot p(d_k)$, який представляє реберну метрику.

Перший множник характеризує ймовірність спостереження отриманої величини (зашумлені інформаційний символ та біт парності)

$$x_{\xi_{n=k}} = x_{\xi_k} \quad (2.11)$$

для даної пари станів s' та s , тоді як другий множник визначає апіорну ймовірність вхідного бінарного символу (стан s' відомий, то можливий перехід в стан s):

$$p(s | s') = p(s', s) = p(d_k) = p(d_k = \pm 1). \quad (2.12)$$

Тоді вираз $p(x_{\xi_n} | d_k) \cdot p(d_k)$ набуває вигляду:

$$\gamma_k(s', s) = p(x_{\xi_{j=k}} | s', s) \cdot p(d_k) = p(x_{\xi_k} | d_k = \pm 1) \cdot p(d_k = \pm 1). \quad (2.13)$$

Запишемо відношення функцій правдоподібності у логарифмічному вигляді [18]:

$$\lambda(d_k) = \ln \left(\frac{p(d_k = +1)}{p(d_k = -1)} \right). \quad (2.14)$$

Розв'язком рівняння (2.14) відносно $p(d_k = +1)$ буде вираз:

$$p(d_k = +1) = \frac{\exp(\lambda(d_k))}{1 + \exp(\lambda(d_k))} = \frac{1}{1 + \exp(-\lambda(d_k))}. \quad (2.15)$$

Другий співмножник з рівняння (2.13) визначає два випадки залежно від знаку інформаційного біта d_k . Тоді з використанням формули (2.7) можна отримати рівняння [18]:

$$p(d_k = +1) = \left(\frac{\exp(-0,5 \cdot \lambda(d_k))}{1 + \exp(-\lambda(d_k))} \right) \cdot \exp(+0,5 \cdot \lambda(d_k)), \quad (2.16)$$

$$\begin{aligned} p(d_k = -1) &= 1 - p(d_k = +1) = \frac{\exp(-\lambda(d_k))}{1 + \exp(-\lambda(d_k))} = \\ &= \frac{\exp(-0,5 \cdot \lambda(d_k))}{1 + \exp(-\lambda(d_k))} \cdot \exp(-0,5 \cdot \lambda(d_k)) \end{aligned} \quad (2.17)$$

Проаналізувавши формули (2.15)–(2.17), можна представити апіорну ймовірність $p(d_k) = p(d_k = \pm 1)$ для вхідного бінарного символу у формі:

$$\begin{aligned} p(d_k = \pm 1) &= \frac{\exp(\pm \lambda(d_k))}{1 + \exp(\pm \lambda(d_k))} = \frac{\exp(d_k \cdot \lambda(d_k))}{1 + \exp(d_k \cdot \lambda(d_k))} = \\ &= \left(\frac{\exp(-0,5 \cdot \lambda(d_k))}{1 + \exp(-\lambda(d_k))} \right) \cdot \exp(d_k \cdot 0,5 \cdot \lambda(d_k)) = A_k \cdot \exp(d_k \cdot 0,5 \cdot \lambda(d_k)). \end{aligned} \quad (2.18)$$

Детектування послідовності x_{ξ_k} може відбутися разом з однією з подій $d_k = \pm 1$. Тоді, використовуючи теорему повної імовірності, можна отримати:

$$\begin{aligned}
p(x_{\xi_k}) &= p(x_{\xi_k}, d_k = +1) + p(x_{\xi_k}, d_k = -1) = \\
&= p(x_{\xi_k} | d_k = +1) \cdot p(d_k = +1) + p(x_{\xi_k} | d_k = -1) \cdot p(d_k = -1).
\end{aligned} \tag{2.19}$$

Врахуємо, що після передавання даних каналом з адитивним білим гаусівським шумом значення $\lambda(x_{\xi_k} | d_k)$ визначається у формі:

$$\begin{aligned}
\lambda(x_{\xi_k} | d_k) &= \ln \left(\frac{p(x_{\xi_k} | d_k = +1)}{p(x_{\xi_k} | d_k = -1)} \right) = \ln \left(\frac{\frac{1}{\sigma \cdot \sqrt{2\pi}} \cdot \exp \left(\frac{-E_c \cdot (x_{\xi_k} - 1)^2}{2 \cdot \sigma^2} \right)}{\frac{1}{\sigma \cdot \sqrt{2\pi}} \exp \left(\frac{-E_c \cdot (x_{\xi_k} + 1)^2}{2 \cdot \sigma^2} \right)} \right) = \\
&= \frac{2}{\sigma^2} \cdot E_c \cdot x_{\xi_k} = 4R \cdot \frac{E_b}{N_0} \cdot x_{\xi_k} = \lambda_{\text{кан}} \cdot x_{\xi_k}.
\end{aligned} \tag{2.20}$$

Тоді можна отримати рівняння [3]:

$$\begin{aligned}
p(x_{\xi_k} | d_k = +1) &= p(x_{\xi_k} | d_k = -1) \cdot \exp(\lambda_{\text{кан}} \cdot x_{\xi_k}) = \\
&= \left(\frac{p(x_{\xi_k}) - p(x_{\xi_k} | d_k = +1) \cdot p(d_k = +1)}{p(d_k = -1)} \right) \cdot \exp(\lambda_{\text{кан}} \cdot x_{\xi_k}).
\end{aligned} \tag{2.21}$$

Розв'язавши рівняння (2.19) відносно умовної ймовірності $p(x_{\xi_k} | d_k = +1)$, отримаємо:

$$p(x_{\xi_k} | d_k = +1) = \frac{\frac{p(x_{\xi_k})}{p(d_k = -1)} \cdot \exp(\lambda_{\text{кан}} \cdot x_{\xi_k})}{1 + \frac{p(d_k = +1)}{p(d_k = -1)} \cdot \exp(\lambda_{\text{кан}} \cdot x_{\xi_k})}. \tag{2.22}$$

Підставивши до рівняння (2.22) вираз (2.18), можна отримати [14]:

$$p(x_{\xi_k} | d_k = +1) = \frac{p(x_{\xi_k}) \cdot (1 + \exp(-\lambda(d_k))) \cdot \exp(-0,5 \cdot \lambda_{кан} \cdot x_{\xi_k})}{1 + \exp(-\lambda(d_k) - \lambda_{кан} \cdot x_{\xi_k})} \times \exp(+0,5 \cdot \lambda_{кан} \cdot x_{\xi_k}) \quad (2.23)$$

Аналогічним чином можна подати рівняння для умовної ймовірності $p(x_{\xi_k} | d_k = -1)$. Результат представлено у такій формі:

$$p(x_{\xi_k} | d_k = -1) = \frac{p(x_{\xi_k}) \cdot (1 + \exp(-\lambda(d_k))) \cdot \exp(-0,5 \cdot \lambda_{кан} \cdot x_{\xi_k})}{1 + \exp(-\lambda(d_k) - \lambda_{кан} \cdot x_{\xi_k})} \times \exp(-0,5 \cdot \lambda_{кан} \cdot x_{\xi_k}) \quad (2.24)$$

На основі рівнянь (2.23) та (2.24) умовна ймовірність $p(x_{\xi_k} | d_k = \pm 1)$ у загальному вигляді представлена рівнянням:

$$\begin{aligned} p(x_{\xi_k} | d_k) &= p(x_{\xi_k} | d_k = \pm 1) = \\ &= \frac{p(x_{\xi_k}) \cdot (1 + \exp(-\lambda(d_k))) \cdot \exp(-0,5 \cdot \lambda_{кан} \cdot x_{\xi_k})}{1 + \exp(-\lambda(d_k) - \lambda_{кан} \cdot x_{\xi_k})} \times \\ &\times \exp(\pm 0,5 \cdot \lambda_{кан} \cdot x_{\xi_k}) = B_k \cdot \exp(d_k \cdot 0,5 \cdot \lambda_{кан} \cdot x_{\xi_k}) \end{aligned} \quad (2.25)$$

Розглянувши серію незалежних подій (отримання інформаційного біта $d_{k,1}$ та бітів парності $d_{k,v}$) між станами ґратки s' і s для послідовності інформаційного біта та n бітів парності для одного декодера, згідно теореми множення ймовірностей можна отримати:

$$\begin{aligned}
p(x_{\xi_k} | d_k) &= p(x_{\xi_{k,1}} | d_{k,1}, x_{\xi_{k,2}} | d_{k,2}, \dots, x_{\xi_{k,n}} | d_{k,n}) = \\
&= p(x_{\xi_{k,1}} | d_{k,1}) \cdot p(x_{\xi_{k,2}} | d_{k,2}) \cdot \dots \cdot p(x_{\xi_{k,n}} | d_{k,n}) = \\
&= p(x_{\xi_{k,1}} | d_{k,1}) \cdot \prod_{v=2}^n p(x_{\xi_{k,v}} | d_{k,v}).
\end{aligned} \tag{2.26}$$

Аналізуючи рівняння (2.25) та (2.26), можна отримати:

$$\begin{aligned}
p(x_{\xi_k} | d_k) &= C_k \cdot \left(\exp(d_{k,1} \cdot 0,5 \cdot \lambda_{кан} \cdot x_{\xi_{k,1}}) \cdot \prod_{v=2}^n \exp(d_{k,v} \cdot 0,5 \cdot \lambda_{кан} \cdot x_{\xi_{k,v}}) \right) = \\
&= C_k \cdot \exp(d_{k,1} \cdot 0,5 \cdot \lambda_{кан} \cdot x_{\xi_{k,1}}) \cdot \exp\left(0,5 \cdot \lambda_{кан} \cdot \sum_{v=2}^n d_{k,v} \cdot x_{\xi_{k,v}}\right) = \\
&= C_k \cdot \exp\left(0,5 \cdot \lambda_{кан} \cdot \left(d_{k,1} \cdot x_{\xi_{k,1}} + \sum_{v=2}^n d_{k,v} \cdot x_{\xi_{k,v}}\right)\right).
\end{aligned} \tag{2.27}$$

Для спрощення приймемо $A_k \cdot C_k = 1$. Тоді транзитна метрика буде представлена у вигляді [14]:

$$\gamma_k(s', s) = \exp\left(0,5 \cdot \left(d_{k,1} \cdot \lambda(d_k) + \lambda_{кан} \cdot \left(d_{k,1} \cdot x_{\xi_{k,1}} + \sum_{v=2}^n d_{k,v} \cdot x_{\xi_{k,v}}\right)\right)\right). \tag{2.28}$$

Для рекурсивного систематичного кодера логарифм від Λ дозволяє отримати метрику шляху для алгоритму декодування Вітербі з м'яким виходом у такому вигляді:

$$M_k(s) = \ln \Lambda = \ln p(x_{\xi_0}^{n-1}, \{s_0, \dots, s_{n-1}\}) + \ln \gamma_k(s', s), \tag{2.29}$$

або, виконавши підстановку, можна отримати вираз:

$$\begin{aligned}
M_k(s) &= M_{k-1}(s') + 0,5 \cdot \left(d_{k,1} \cdot \lambda(d_k) + \lambda_{\text{кан}} \cdot \left(d_k \cdot x_{\xi_{k,1}} + \sum_{v=2}^n d_{k,v} \cdot x_{\xi_{k,v}} \right) \right) = \\
&= M_{k-1}(s') + \gamma_k^*(s', s),
\end{aligned} \tag{2.30}$$

де $M_{k-1}(s')$ – поточна метрика у вершині трелліс-діаграми.

Після логарифмування максимум функції буде в тому самому стані на трелліс, але змінить значення, що не впливає на роботу алгоритму декодування. Значеннями $\ln(A_k \cdot C_k)$ можна знехтувати, оскільки вони не впливають на розрахунки (константа для кожного символу на трелліс). Отже, в задачі оптимізації суттєвий вплив має значення реберної метрики, тобто

$$\gamma_k^*(s', s) \rightarrow \max. \tag{2.31}$$

Враховуючи те, що метрика ребра додається на кожному кроці, то можна отримати вираз, який потрібно максимізувати, тобто вибрати найбільшу

метрику шляху $\max_{d_k=\pm 1}^{(s', s)} (M_k(s))$ [14].

Отже, алгоритм декодування передбачає м'яку оцінку надійності для послідовності бітів (дає змогу отримати найбільш правдоподібний шлях на ґратці, який складається з виживших підшляхів), проходячи певну кількість етапів трелліс в режимі ковзного вікна розміром δ . Зазвичай довжина вікна встановлюється рівною $5 \cdot K = 5 \cdot (m + 1)$. Якщо декілька підшляхів, які ведуть до того самого стану на трелліс, мають однакову метрику шляху $M_k(s)$, використовується правило кидання монети для розв'язання невизначеності, тобто вибирається випадковим чином один підшлях з метрикою $M_k(s)$ [11].

Модифікація даного алгоритму для більш точної оцінки надійності бінарного символу виконується відповідно до формули Хагенауера [24]. Розрахуємо різницю у стані k на трелліс між метриками вижившого шляху $M_k^{(1)}(s)$ та відкинутого $M_k^{(2)}(s)$ за наступною формулою (рисунк 2.6):

$$\Delta_k^{0,\dots,\delta} = M_k^{(1)}(s) - M_k^{(2)}(s). \quad (2.32)$$

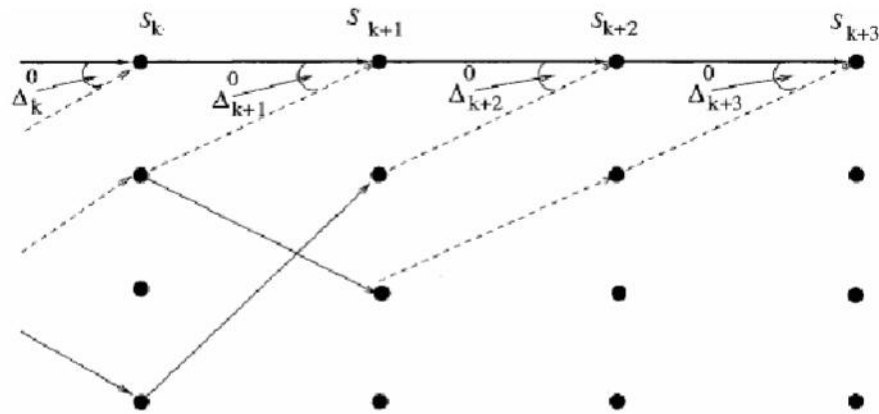


Рисунок 2.6 – Визначення значень різниць $\Delta_k^{0,\dots,\delta}$

Якщо шлях вважається вижившим, то він правильний, тобто

$$p(\text{true}) = \frac{p(\text{виживший шлях})}{p(\text{виживший шлях}) + p(\text{відкинутий шлях})}. \quad (2.33)$$

$$p(\text{true}) = \frac{e^{M_k^{(1)}(s)}}{e^{M_k^{(1)}(s)} + e^{M_k^{(2)}(s)}} = \frac{e^{M_k^{(1)}(s) - M_k^{(2)}(s)}}{1 + e^{M_k^{(1)}(s) - M_k^{(2)}(s)}} = \frac{e^{\Delta_k^{0,\dots,\delta}}}{1 + e^{\Delta_k^{0,\dots,\delta}}}. \quad (2.34)$$

Таким чином, якщо розглянути відношення для знаходження м'яких рішень у логарифмічній формі, то можна отримати:

$$\ln\left(\frac{p(true)}{p(false)}\right) = \ln\left(\frac{p(true)}{1 - p(true)}\right) = \Delta_k^{0, \dots, \delta}. \quad (2.35)$$

Дане значення пропонується використовувати для пошуку апостеріорних рішень. За *mod 2* порівнюються два конкуруючі (виживший та відкинутий) шляхи, які проходять через Δ_k на певному етапі ґратки. Якщо бінарний символ вижившого шляху співпадає з відкинутим (результат дорівнює 0), то на відповідному місці ставиться ∞ , інакше (1) – Δ . Далі складається матриця надійностей рішень та знаходиться мінімум на кожному етапі трелліс. Розрахунок м'яких рішень можна представити у вигляді [18]:

$$\lambda_v(d_k) \approx d_k \cdot \min_{\substack{i=0 \dots \delta \\ d_k \neq d_k^i}} \Delta_k^i. \quad (2.36)$$

У двонаправленому алгоритмі у першій частині декодування відбувається так само, як у звичайному алгоритмі Вітербі з м'яким виходом, який вибирає найбільш імовірну кодову послідовність, що відповідає шляху прокладеному на трелліс-діаграмі "вперед", а у другій частині алгоритм виконується "назад". Відповідно м'які апостеріорні рішення потрібно знаходити на прямому та зворотному проходах на трелліс. Отже, загальну формулу можна представити у такому вигляді [24]:

$$\lambda_{BiV}(d_k) = \begin{cases} \lambda_v^{\rightarrow}(d_k), & \text{якщо } |\lambda_v^{\rightarrow}(d_k)| \geq |\lambda_v^{\leftarrow}(d_k)|; \\ \lambda_v^{\leftarrow}(d_k), & \text{якщо } |\lambda_v^{\leftarrow}(d_k)| > |\lambda_v^{\rightarrow}(d_k)|, \end{cases} \quad (2.37)$$

де $\lambda_v^{\rightarrow}(d_k)$, $\lambda_v^{\leftarrow}(d_k)$ – апостеріорні рішення, які знайдено відповідно за прямим та зворотним алгоритмом на сегменті трелліс з ковзним вікном розміром δ .

Інший варіант: виконати рух вперед та назад до кінця трелліс та обчислити апостеріорні виходи по кожному біту у вигляді:

$$\lambda_{BiV}(d_k) = \min_{d_k=+1}^{(s',s)}(M_k(s)) - \min_{d_k=-1}^{(s',s)}(M_k(s)). \quad (2.38)$$

Для фрейму з великою кількістю бінарних символів може виникнути проблема числового бар'єру внаслідок множення великих чисел. Тому метрики шляху назад та вперед нормують у вигляді:

$$M_k'(s) = M_k(s) - \psi, \quad (2.39)$$

де ψ – деякий поріг значень.

Метрики можуть бути нормалізовані шляхом ділення на максимальне значення метрики $M_k(s)$ на поточному кроці.

Ітеративну формулу обчислення апостеріорних рішень на q ітераціях роботи турбо-декодера можна записати наступним чином [3]:

$$\lambda_{апост}^{(i)}(d_k) = \lambda_{ап}^{(i)}(d_k) + \lambda_{кан.} \cdot x_{\xi_k} + \lambda_{зовн}^{(i)}(d_k), \quad (2.40)$$

$$\lambda_{ап}^{(i)}(d_k) = \lambda_{зовн}^{(i-1)}(d_k), i = \overline{1, 2q}. \quad (2.41)$$

Слід зазначити, що зазвичай, коли $i = 1$, то дані вважаються рівноймовірними, тобто апіорне значення $\lambda_{ап}(d_k) = 0$.

2.5 Висновки

Розглянуті основні аспекти турбо-кодування / декодування даних. Представлено процес декодування у вигляді оптимізаційної задачі пошуку максимально правдоподібної послідовності бінарних символів. Дану задачу розв'язано з використанням групи м'яких алгоритмів декодування Вітербі. У наступному розділі необхідно дослідити ефективність роботи розробленого турбо-кодека у складі системи цифрового зв'язку.

3 ЕКСПЕРИМЕНТАЛЬНІ ДОСЛІДЖЕННЯ ЕФЕКТИВНОСТІ РОБОТИ ТУРБО-КОДЕКА

3.1 Вибір мови програмування

Мови програмування – це інструменти, які дозволяють розробникам створювати програми і взаємодіяти з комп'ютерами на високому рівні. Історія мов програмування почалася з низькорівневого машинного коду, який було важко написати і зрозуміти. Пізніше з'явилися перші мови програмування високого рівня, які дали можливість розробникам працювати з абстрактними поняттями і були легшими для розуміння людей. Однією з найвідоміших мов програмування є Fortran, яка була створена в 1957 році для використання в наукових дослідженнях та інженерних обчисленнях. У 1960-х роках з'явилися Cobol і Basic, які були націлені на ширшу аудиторію, а в 1970-х років C стала найпоширенішою мовою програмування. Потім з'явилися інші мови, такі як C++, Java, Python, М, JavaScript, кожна зі своїми характеристиками та галузями використання. Сьогодні мови програмування є важливими інструментами для розробки програмного забезпечення, веб-сайтів, мобільних додатків і багатьох інших різноманітних програм [25].

Впродовж історії обчислювальної техніки еволюція мов програмування означала зміни в обчислювальному середовищі, способі мислення програмістів і самих методах програмування. Слід відмітити, що мова М середовища MatLab найчастіше використовується у вищих навчальних закладах світу для розробки наукових програм, швидких інженерно-математичних розрахунків, навчання основам програмування. Вона входить до числа найпопулярніших ресурсів на GitHub і Stack Overflow (рисунок 3.1).

бібліотеки та інструменти, які автоматично налаштовують алгоритми для конкретних архітектур процесорів, можна зменшити обчислювальну складність і час роботи програми.

Отже, згідно з поставленими умовами, програмне забезпечення можна розробити на мові програмування М, оскільки вона володіє достатньою кількістю спеціалізованих бібліотек для математичних розрахунків та може бути доповнена реалізацією деяких методів за допомогою C++ та Python.

3.2 Опис роботи програми

Вхідними даними є інформаційний блок, параметри компонентних кодів, вид інтерлівера, критерій зупинки ітеративного декодера, крок зміни значення E_b/N_0 , мінімальне та максимальне значення E_b/N_0 . Важливі вихідні дані – послідовність символів на виході, значення $BER = f(E_b/N_0)$ та відповідно графічно побудована залежність.

При роботі з програмним забезпеченням користувач має ввести та вибрати потрібні для роботи вхідні дані у визначених для цього місцях інтерфейсу програми. Потім натиснути на кнопку-перемикач $BER = f(E_b / N_0)$ або $BER = f(q)$ у нижній панелі результатів для початку роботи програмних модулів. Програма починає видавати докладну статистику у робочому вікні. Після закінчення процесу передавання даних можна переглянути графічні зображення в залежності від вибору користувача. Виконавши всі потрібні операції, користувач може зберегти потрібні результати, надрукувати їх або закрити вікно програми, завершивши роботу з нею.

Основні функції, які розроблені у програмі:

- 1) `turbo_encode(cod, ...)`, `turbo_decode(dec...)` – кодування та декодування, де `cod`, `dec` – компонентний алгоритм кодування / декодування;
- 2) `BPSK(...)`, `de_BPSK(...)` – проведення модуляції / демодуляції;
- 3) `AWGN_channel(...)` – моделювання каналу зв'язку з нормально розподіленим адитивним шумом;
- 4) `result(...)`, `save(...)`, `print(...)` – обчислення результатів, їх збереження та друк.

Основні технічні вимоги та мінімальні системні вимоги до програми: операційна система Windows 7 або новіша; наявність клавіатури та миші; процесор Core i5 (Ryzen 5) або кращий; оперативна пам'ять – 8 Гб; відеопам'ять – 512 Мб; жорсткий диск – 500 Гб і більше; встановлене програмне забезпечення MatLab.

3.3 Експерименти

Щоб знайти експериментальні залежності

$$BER = f\left(\frac{E_b}{N_0}\right), \quad (3.1)$$

які характеризують ефективність декодування, використаємо імітаційне моделювання методом Монте-Карло, тобто багаторазове випробування системи зі статистичною обробкою отриманих даних для розв'язання задач аналізу,

синтезу та експлуатації згорткових турбо-кодів у комп'ютерних системах різного функціонального призначення [4, 17, 26]. Загальна імітаційна модель системи зв'язку, яка була використана для проведення моделювання, показана на рисунку 3.2.



Рисунок 3.2 – Структура експерименту

Експеримент 1. Проводиться дослідження робочих характеристик турбо-декодера з використанням звичайного та двонаправленого алгоритмів декодування Вітербі з м'яким виходом. Використовується пакет даних з виходу генератора псевдовипадкової послідовності розміром 1024 біта, турбо-код із двома компонентними рекурсивними систематичними згортковими кодерами з поліноміальним генератором $(7, 5)_8$, швидкість коду $R = 1/3$, псевдовипадковий інтерлівер із генерацією рівномірно розподілених випадкових величин за

алгоритмом Парка-Міллера, термінація кодера відбувається за допомогою хвостової послідовності на $m = 2$ бінарних символи, максимум 10 ітерацій декодування, бінарний модем, модель каналу з адитивним білим гаусівським шумом. Моделювання проводиться від 0 до 3 дБ з кроком 0,25 дБ. Умовою закінчення процесу моделювання для кожного значення E_b/N_0 є час (не більше 1 години на крок) та кількість пакетів, в яких наявні помилки у бінарних символах (не менше 10 фреймів). Результати даного експерименту показані на рисунку 3.3.

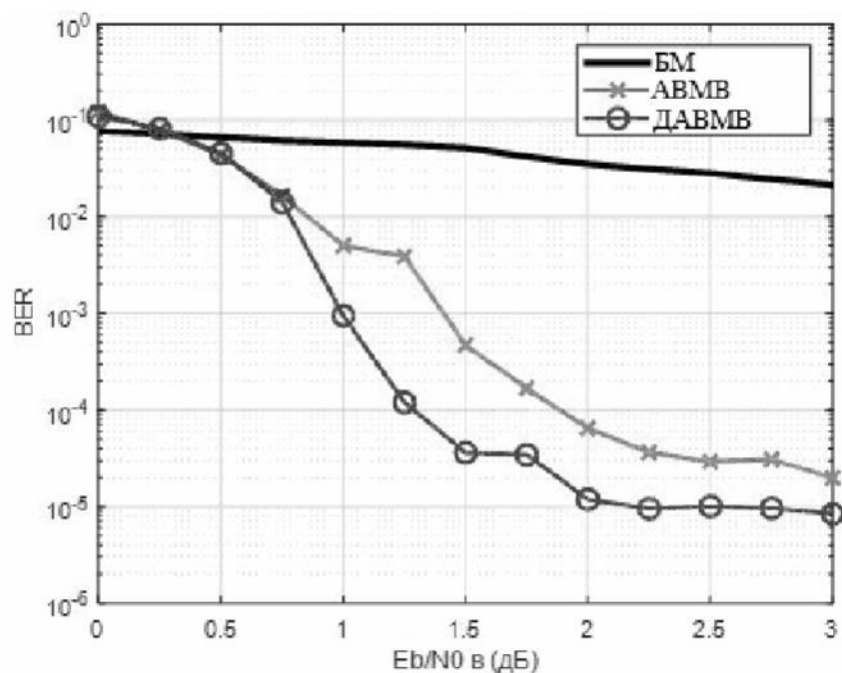


Рисунок 3.3 – Результати декодування даних

Можна побачити, що дані алгоритми турбо-декодування дозволяють досягти енергетичного виграшу порівняно з некодованою кривою бінарного модему (БМ). Крім того, двонаправлений алгоритм (ДАВМВ) дає виграш 0,43...0,65 дБ, починаючи зі значення $E_b/N_0 = 1$ дБ для $BER = 10^{-3}$ - 10^{-5} , порівняно з звичайним алгоритмом Вітербі (АВМВ). Із рисунку 3.3 також

видно, що спостерігається різкий спад у діапазоні низьких значень $E_b/N_0 \leq 2$ дБ, але зі збільшенням цієї величини помітно поступове досягнення “плато насичення”. Показники BER для даних алгоритмів декодування наведені у таблиці 3.1.

Таблиця 3.1 – Показники BER залежно від значення E_b/N_0

E_b/N_0 , дБ	BER	
	АВМВ	ДАВМВ
0,00	$1,241 \cdot 10^{-1}$	$1,121 \cdot 10^{-1}$
0,50	$4,332 \cdot 10^{-2}$	$4,332 \cdot 10^{-2}$
1,00	$5,235 \cdot 10^{-3}$	$9,381 \cdot 10^{-4}$
1,50	$4,347 \cdot 10^{-4}$	$3,614 \cdot 10^{-5}$
2,00	$6,248 \cdot 10^{-5}$	$1,121 \cdot 10^{-5}$
2,50	$2,766 \cdot 10^{-5}$	$9,981 \cdot 10^{-6}$
3,00	$1,536 \cdot 10^{-5}$	$8,369 \cdot 10^{-6}$

Експеримент 2. Виконаємо передавання кольорового зображення “lena.jpg” (768 фреймів по 4096 бітів) в умовах високих значень потужностей шумів ($E_b/N_0 = 1,5$ дБ) з використанням ДАВМВ. Чисельні показники наведені в таблиці 3.2, де BEN / SEN – кількість бітових та символічних помилок, а BER / SER – коефіцієнт бітових та символічних помилок. Графічно дані відображені на графіку з рисунку 3.4.

Зі збільшенням кількості ітерацій ефективність декодера стає кращою, оскільки зменшується кількість помилок на зображенні. Можна побачити, що для того, щоб отримати хороші результати при умові значної потужності

шуму, необхідно виконати багато ітерацій декодування $Q > 20$ ($BER \approx 10^{-4}$, $SER \approx 10^{-3}$).

Таблиця 3.2 – Чисельні показники експерименту

E_b/N_0 , дБ	Кількість ітерацій	BEN	SEN	BER	SER
1,5 дБ	0	576264	187142	$1,832 \cdot 10^{-1}$	0,9519
	1	234512	122242	$7,454 \cdot 10^{-2}$	0,6217
	4	37624	26418	$1,201 \cdot 10^{-2}$	0,1343
	12	2277	1487	$7,238 \cdot 10^{-4}$	0,0075
	20	611	363	$1,942 \cdot 10^{-4}$	0,0018

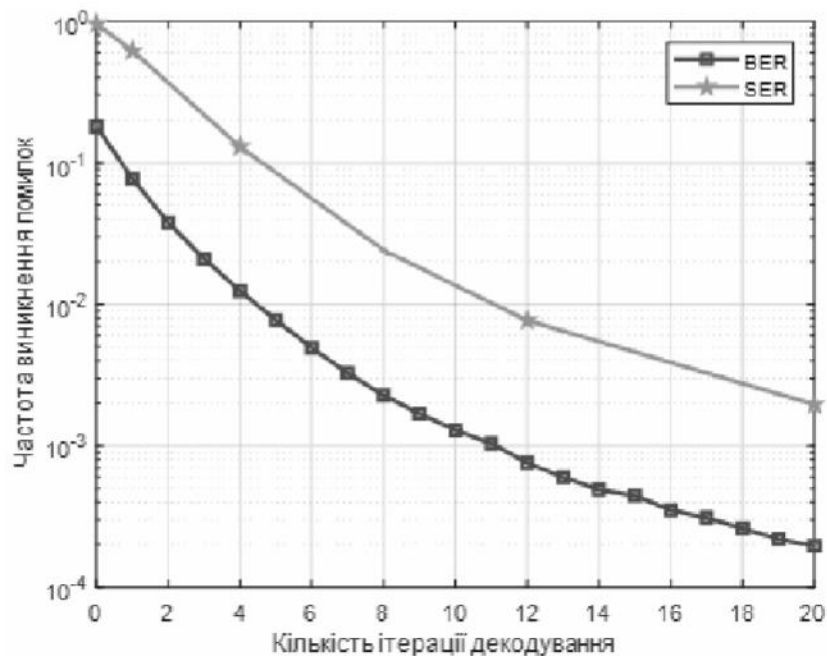


Рисунок 3.4 – Результати декодування зображення

Результати в мультимедійному вигляді показані на рисунках 3.5–3.8, де в лівому верхньому куті вказано кількість ітерацій декодування q .

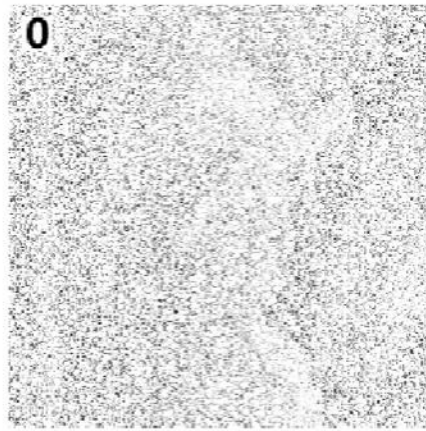


Рисунок 3.5 – Зображення без кодування, BPSK-модем



Рисунок 3.6 – Зображення після 1 ітерації декодування



Рисунок 3.7 – Зображення після 4 ітерацій декодування



Рисунок 3.8 – Зображення після 20 ітерацій декодування

Зі збільшенням кількості ітерацій декодування якість зображення покращується (кількості помилок у бітах та символах зменшуються). Отже, результати експериментів свідчать про те, що ДАВМВ дозволяє досягти хорошої коригуючої здатності при низьких відношеннях сигнал/шум для турбо-кодів.

Двонаправлена версія виграє за ефективністю у класичного алгоритму. Це робить даний алгоритм перспективним інструментом для використання у комунікаційних системах в умовах обмежених ресурсів. Складність декодування одного біта ДАВМВ більша за АВМВ, навіть враховуючи обчислення метрики ребра один раз.

Одержані результати можуть бути корисними у ході поглибленого аналізу турбо-кодів та обґрунтованого вибору елементної бази при їх апаратно-програмній реалізації, з урахуванням специфічних вимог до систем передавання цифрових даних.

3.4 Висновки

Розроблена програма дозволила виконати експериментальні дослідження ефективності роботи турбо-кодека та складових алгоритмів турбо-декодування. Визначено, що ДАВМВ працює краще за АВМВ, хоча потребує більших обчислювальних ресурсів. Для передавання мультимедійних даних в умовах сильного шуму необхідно задати достатню кількість ітерацій декодування.

ВИСНОВКИ

Аналіз методів завадостійкого кодування показав, що найбільш перспективним є згортковий турбо-код, який дозволяє працювати в умовах сильного шуму. Для досягнення максимальної ефективності використовуються турбо-декодери з м'яким виходом, які потребують розробки оптимізованих обчислювальних алгоритмів декодування. У роботі розглядаються алгоритми пошуку максимально правдоподібної послідовності, представлено їх математичний апарат.

Розроблено програмне забезпечення, яке дозволяє протестувати процес передавання даних із використанням відповідних компонентних турбо-декодерів та алгоритмів. Отримано експериментальні криві частоти виникнення помилок від значення сигнал/шум, які показують, що ДАВМВ ефективніший за АВМВ, але обчислювально складніший. Отже, отримані результати підтверджують ефективність розглянутої групи алгоритмів турбо-декодування.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Інформаційні основи побудови телекомунікаційних мереж / В.В. Казимир, В.В. Литвинов, С.В. Зайцев, та С.М. Шкарлет. Чернігів: Чернігівський державний технологічний університет, 2013. С. 82–142.
2. Основи техніки передавання інформації / Р. Н. Кветний, А. Я. Кулик М. М. Компанець, С. Г. Кривоғубченко. Вінниця: ВНТУ, 2001. 364 с.
3. Channel Coding: Theory, Algorithms, and Applications / D. Declerq et al. Academic Press Library in Mobile and Wireless Communications, 2014. 690 p.
4. Mishra D., Jit S., Dasgupta K. S. Hybrid Concatenated Convolutional Code for Deep Space Mission. *International Journal of Computer Applications*. New York, 2012. Vol. 47. pp. 33–38.
5. Xia X.-G. Understanding Turbo Codes: A Signal Processing Study. *Journal of Information and Intelligence*. 2024. pp. 1–13. URL: <https://arxiv.org/pdf/2308.13148>. (дата звернення 23.04.2025)
6. Morgos F.L., Cuc A.-M., Grava C. Performance Analysis of Turbo Codes, LDPC Codes, and Polar Codes over an AWGN Channel in the Presence of Inter Symbol Interference. *Sensors*. Basel, 2023. №4. 19 p. URL: <https://www.mdpi.com/1424-8220/23/4/1942>. (дата звернення 25.04.2025)
7. Le V.H.S. Design of Next-Generation Tbps Turbo Codes. Ecole Nationale Supérieure Mines-Télécom Atlantique, 2021. 131 p.
8. Ivanov Yu.Yu., Kovtun V.V. Crypto Coding System Based on the Turbo Codes with Secret Keys. *ICT Express*. 2024. Vol. 10 (2). pp. 330–335.
9. Іванов Ю.Ю. Експериментальне дослідження завадостійкості турбо-кодів: числові оцінки та імітаційне моделювання нового субоптимального

алгоритму PL-log-MAP. *Вісник Вінницького політехнічного інституту*. Вінниця: ВНТУ, 2016. № 5. С. 76-84.

10. Евристичні алгоритми дискретної оптимізації / Т.В. Пасічний, Ю.Ю. Іванов, С.Г. Кривогубченко та інші. *LIV Всеукраїнська науково-технічна конференція підрозділів Вінницького національного технічного університету*. Вінниця, 24-27.03.2025. URL: <https://conferences.vntu.edu.ua/index.php/all-fksa/all-fksa-2025/paper/view/23530/19764>. (дата звернення 16.04.2025)

11. Hanzo L., Liew T. H., Yeap B. L. Turbo Coding, Turbo Equalisation and Space-Time Coding for Transmission over Wireless Channels. Southampton: Department of Electronics and Computer Science, 2002. 746 p.

12. Gracie K., Hamon M.-H. Turbo and Turbo-Like Codes: Principles and Applications in Telecommunications. *Proceedings of the IEEE*. 2007. Vol. 95. № 5. pp. 1228–1254.

13. Boutillon E., Douillard C., Montorsi G. Iterative Decoding of Concatenated Convolutional Codes: Implementation Issues. *Proceedings of IEEE*. 2007. Vol. 95. pp. 1201–1227.

14. Channel Coding in Communication Networks. From Theory to Turbocodes / C. Berrou, C. Douillard, M. Jezequel, A. Picart. Chippenham: ISTE, 2007. 437 p.

15. Johnson S.J. Iterative Error Correction. Turbo, Low-Density Parity-Check and Repeat-Accumulate Codes. New York: Cambridge University Press, 2009. 356 p.

16. Berrou C., Glavieux A. Near Optimum Error Correcting Coding and Decoding: Turbo-codes. *IEEE Transactions on Communications*. 1996. Vol. 44. pp. 1261–1271.

17. Особливості оцінювання параметрів процесу передавання даних із використанням турбо-кодів / Р.Н. Кветний, Ю.Ю. Іванов, С.Г. Кривогубченко, О.В. Стукач. *Метрологія та прилади*. Київ, 2017. № 3. С. 25–32.

18. Woodard J., Hanzo L. Comparative Study of Turbo Decoding Techniques: An Overview. *IEEE Transactions on Vehicular Technology*. 2000. Vol. 49. pp. 2208–2233.
19. Ivanov Y., Kulyk A., Krivogubchenko S. A Viterbi Algorithm as a Key to Decoding Turbo-Code. *Nauka i studia*. 2012. № 11 (56). pp. 60–65.
20. Кулик А.Я., Иванов Ю.Ю. Апаратна реалізація декодера SOVA з “м’яким” 3-бітовим квантованим виходом. *Цифрові технології*. Одеса: ОНАЗ ім. О.С. Попова, 2012. № 12. С. 15-22.
21. Промислові мережі та інтеграційні технології в автоматизованих системах / О.М. Пупена, І.В. Ельперін, Н.М. Луцька та інші. 2021. 522 с.
22. Robertson P., Villebrun P., Hoeher P. Optimal and Sub-Optimal Maximum A Posteriori Algorithms Suitable for Turbo Decoding. *European Transactions on Telecommunications*. 1997. Vol. 8. pp. 119–125.
23. Модифікований алгоритм турбо-декодування для турбо-кодів із повторенням / О. Овчаров, С. Зайцев, С. Лівенцев, О. Петрова. *Науковий журнал “Зв’язок”*. 2011. № 4. С. 38–46.
24. Bi-directional SOVA Decoding for Turbo-Codes / J. Chen, M.C. Fossorier, S. Lin, C. Xu. *IEEE Communications Letters*. 2001. Vol. 4 (12). pp. 405–407.
25. Jiang Y. A Practical Guide to Error-Control Coding Using MATLAB. London: Artech House, 2010. 293 p.
26. Simulating the Operation of Turbo Codes through the Monte Carlo Method, Comparison between MATLAB, C and C# / H. Balta, A. Isar, D. Isar, M. Balta. *Advances in Intelligent and Soft Computing*. Springer Berlin Heidelberg, 2012. Vol. 133. pp. 1127–1139. URL: https://www.researchgate.net/publication/268366821_Simulating_the_Operation_of_Turbo_Codes_through_the_Monte_Carlo_Method_Comparison_between_MATLAB_C_and_C. (дата звернення 05.05.2025)

ДОДАТКИ

(обов'язковий)

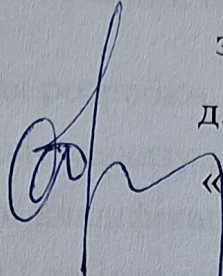
Технічне завдання

ЗАТВЕРДЖУЮ

завідувач кафедри АПТ

д.т.н., проф. Бісікало О. В.

« 23 » 03 2025 р.



ТЕХНІЧНЕ ЗАВДАННЯ

на бакалаврську кваліфікаційну роботу

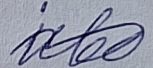
«Розробка автоматизованого модуля для декодування турбо-кодів
на основі пошуку максимально правдоподібної бінарної послідовності»

08-31.БКР.007.02.000 ТЗ

Керівник роботи:

к.т.н., доц. каф. АПТ

Іванов Ю. Ю.

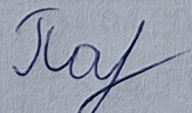


«24» 03 2025 р.

Виконавець:

ст. гр. 1АКІТ-216

Пасічнюк Т. В.



«24» 03 2025 р.

1. Назва та галузь застосування

Наявність завад різної природи зумовлює актуальність розробки методів і засобів, які дозволяють виявляти та коригувати помилки передавання даних. На даний момент одним із найбільш перспективних методів прямої корекції помилок є турбо-код. У даній роботі розглядаються алгоритми декодування на основі пошуку максимально правдоподібної бінарної послідовності.

2. Підстави для розробки

Розробку здійснювати на підставі наказу по університету №97 від 20.03.2025 та завдання до бакалаврської кваліфікаційної роботи, складеного та затвердженого кафедрою «Автоматизації та інтелектуальних інформаційних технологій».

3. Мета та призначення розробки

Метою роботи є розширення функціональних можливостей програмного забезпечення, в якому реалізовано алгоритми турбо-декодування на основі пошуку максимально правдоподібної послідовності бітів, за рахунок розробки модуля для їх детального експериментального дослідження.

4. Етапи виконання робіт

Виконавцем перерахованих у даному розділі етапів є студентка групи 1AKIT-216 Пасічнюк Тетяна Василівна факультету Інтелектуальних інформаційних технологій та автоматизації Вінницького національного технічного університету, а замовником є кафедра автоматизації та інтелектуальних інформаційних технологій.

№	Назва та зміст етапу	Термін виконання	
		початок	закінчення
1	Аналіз предметної області роботи	21.03.2025	04.04.2025
2	Розробка математичної моделі турбо-кодека	05.04.2025	26.04.2025
3	Експериментальні дослідження ефективності роботи турбо-кодека	27.04.2025	16.05.2025
4	Оформлення пояснювальної записки та ілюстративного матеріалу	17.05.2025	27.05.2025

5. Джерела розробки

1. Іванов Ю.Ю. Експериментальне дослідження завадостійкості турбо-кодів: числові оцінки та імітаційне моделювання нового субоптимального алгоритму PL-log-MAP. *Вісник Вінницького політехнічного інституту*. Вінниця: ВНТУ, 2016. № 5. С. 76-84.
2. Le V.H.S. Design of Next-Generation Tbps Turbo Codes. Ecole Nationale Supérieure Mines-Télécom Atlantique, 2021. 131 p.
3. Kovtun V.V., Ivanov Yu.Yu. Crypto Coding System Based on the Turbo Codes with Secret Keys. *ICT Express*. 2024. Vol. 10 (2). pp. 330–335.

6. Показники призначення

Розроблено програмне забезпечення, яке дозволяє виконати необхідні експериментальні дослідження. Основні технічні вимоги та мінімальні системні вимоги до програми: операційна система *Windows 7* або новіша; наявність клавіатури та миші; процесор *Core i5 (Ryzen 5)* або кращий; оперативна пам'ять – 8 ГБ; відеопам'ять – 512 МБ; жорсткий диск – 500 ГБ і більше; встановлене програмне забезпечення *MatLab*.

Вхідні дані: математичний апарат турбо-декодування на основі пошуку максимально правдоподібної бінарної послідовності (розмір пакету n , кількість кодерів v та ітерацій декодування q); послідовності тестових даних, мультимедійні дані. Результати роботи: експериментальні функціональні залежності частоти виникнення помилок; декодоване кольорове зображення.

Студентка гр. 1AKIT-216

(підпис)

Тетяна ПАСТІЧНЮК
(ініціали та прізвище)

Додаток Б
(обов'язковий)

ІЛЮСТРАТИВНА ЧАСТИНА

РОЗРОБКА АВТОМАТИЗОВАНОГО МОДУЛЯ ДЛЯ ДЕКОДУВАННЯ
ТУРБО-КОДІВ НА ОСНОВІ ПОШУКУ МАКСИМАЛЬНО
ПРАВДОПОДІБНОЇ БІНАРНОЇ ПОСЛІДОВНОСТІ

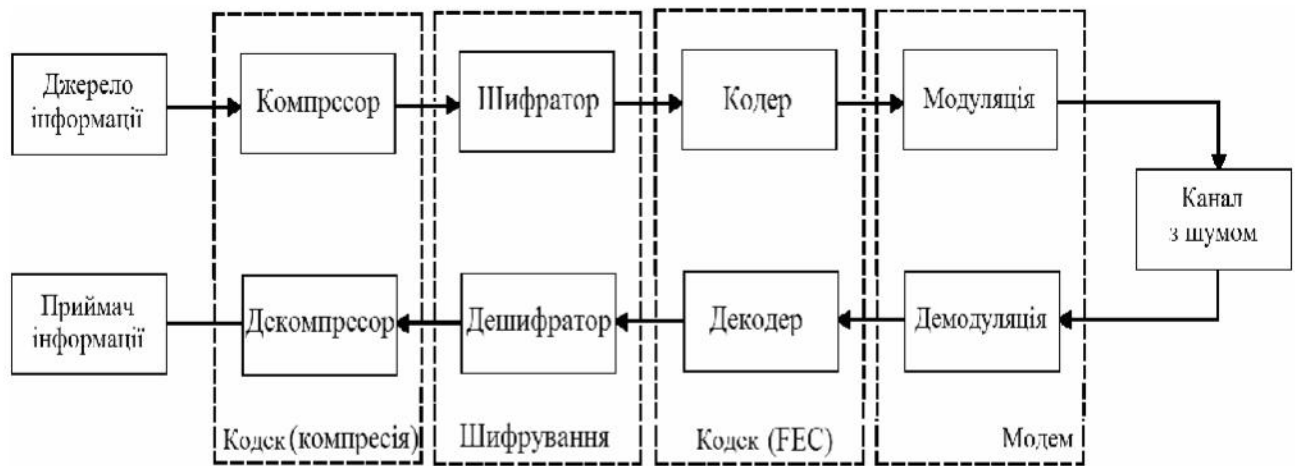


Рисунок Б.1 – Структура системи цифрового зв'язку

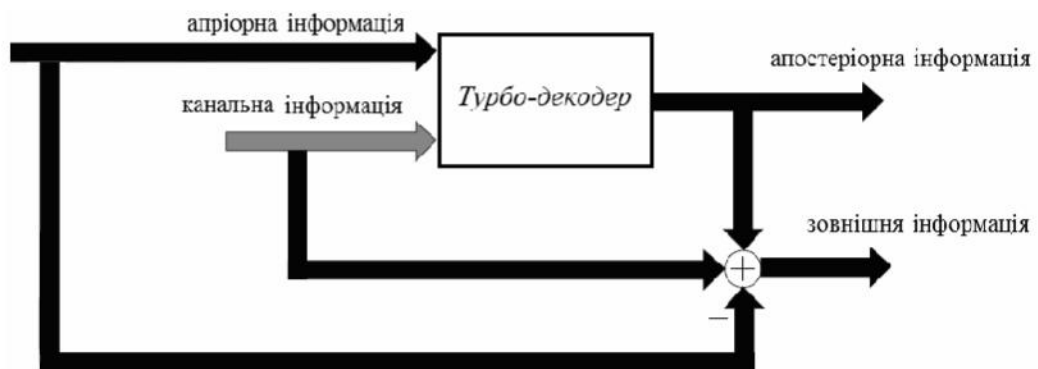


Рисунок Б.2 – Принцип турбо-декодування

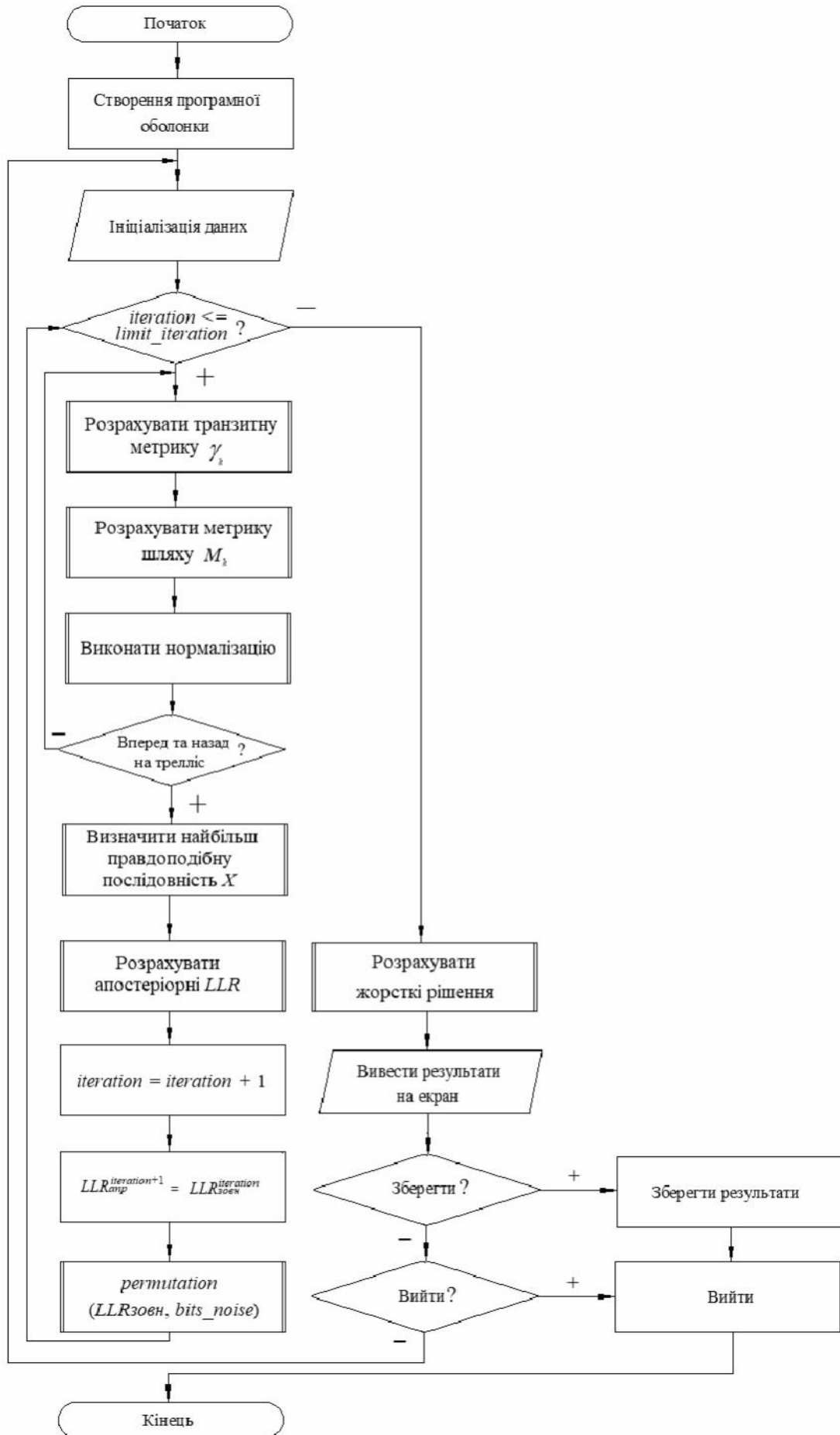


Рисунок Б.3 – Схема програми

Додаток В

(обов'язковий)

Лістинг програмного забезпечення

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function out = vitdec( msg, llr, trl, win )
    enc = trellis2enc( trl );
    if( enc.k == 1 )
        out = vitdec_1N( msg, llr, trl, win );
    else
        out = vitdec_KN( msg, llr, trl, win );
    end
end

function enc = trellis2enc( trl ),
    enc.k = log2( trl.numInputSymbols );
    enc.n = log2( trl.numOutputSymbols );
    enc.r = enc.k / enc.n;
    enc.ksym = trl.numInputSymbols;
    enc.nsym = trl.numOutputSymbols;
    enc.stat = trl.numStates;
    enc.next.states = trl.nextStates + 1;
    enc.next.output = trl.outputs;

    for i = 1:enc.ksym,
        enc.next.binout( :,i ) = 2*de2bi( oct2dec( trl.outputs(:,i) ), enc.n )-1;
    end

    enc.inp = de2bi( oct2dec( [0:enc.ksym-1] ), enc.k, 'left-msb' );
    enc.out = de2bi( oct2dec( [0:enc.nsym-1] ), enc.n, 'left-msb' );
    enc.bininp = 2*enc.inp-1;
end

function out = vitdec_1N( msg, llr, trl, win ),
    if( ~istrellis( trl ) ), error( 'Incorrect input trellis!' ); end;
    if( nargin <= 2 ), error( 'Incorrect number of input args!' ); end;
    if( nargin == 3 ), win = length( llr )+1; end;
    INF = 9e9;
    enc = trellis2enc( trl );
    len = length( llr ) / enc.k;
    win = min( [ win, len ] );
    old = NaN;
    metr = zeros( enc.stat, win+1 ) -INF;
    metr( 1,1 ) = 0;
    surv = zeros( enc.stat, win+1 );

```

```

inpt = zeros( enc.stat, win+1 );
diff = zeros( enc.stat, win+1 );
comp = zeros( enc.stat, win+1 );
inpc = zeros( enc.stat, win+1 );
out  = zeros( size(llr) ) + NaN;
sft  = zeros( size(llr) ) + INF;

for i = 1:len,
    Cur = mod( i-1, win+1 ) +1;
    Nxt = mod( i,   win+1 ) +1;
    buf = msg( i*enc.n:-1:(i-1)*enc.n+1 );
    llb = llr( (i-1)+1:i );
    metr( :,Nxt ) = -INF -INF;

    for s = 1:enc.stat,
        for j = 1:enc.ksym,
            nxt = enc.next.states( s, j );
            bin = enc.next.binout( s,:,j );
            mtr = bin*buf + metr( s,Cur );
            mtr = mtr + enc.bininp( j )*(llb*enc.r)';

            if( metr( nxt,Nxt ) < mtr ),
                diff( nxt,Nxt ) = mtr - metr( nxt,Nxt );
                comp( nxt,Nxt ) = surv( nxt,Nxt );
                inpc( nxt,Nxt ) = inpt( nxt,Nxt );
                metr( nxt,Nxt ) = mtr;
                surv( nxt,Nxt ) = s;
                inpt( nxt,Nxt ) = j-1;
            else
                dif = metr( nxt,Nxt ) - mtr;
                if( dif <= diff( nxt,Nxt ) )
                    diff( nxt,Nxt ) = dif;
                    comp( nxt,Nxt ) = s;
                    inpc( nxt,Nxt ) = j-1;
                end
            end
        end
    end
end

if( i < win ), continue; end;
[ mtr, sur ] = max( metr( :,Nxt ) );
b = i;
clc = mod( Nxt-[1:win], win+1 ) +1;

for j = 1:win,
    inp = inpt( sur, clc(j) );
    out( b ) = inp;
end

```

```
tmp = clc( j );  
cmp = comp( sur, tmp );  
inc = inpc( sur, tmp );  
dif = diff( sur, tmp );  
srv = surv( sur, tmp );  
  
for k = j+1:win+1,  
    if( inp ~= inc ),  
        tmp = dif;  
        idx = b - ( (k-1)-j );  
        sft( idx ) = min( sft(idx), tmp );  
    end  
    if( srv == cmp ), break; end;  
    if( k == win+1 ), break; end;  
    tmp = clc( k );  
    inp = inpt( srv, tmp );  
    inc = inpt( cmp, tmp );  
    srv = surv( srv, tmp );  
    cmp = surv( cmp, tmp );  
end  
sur = surv( sur, clc(j) );  
b = b - 1;  
end  
end  
out = (2*out-1) .* sft;  
end
```

Додаток Г
(обов'язковий)
Протокол перевірки БКР

ПРОТОКОЛ ПЕРЕВІРКИ КВАЛІФІКАЦІЙНОЇ РОБОТИ

69

Назва роботи: Розробка автоматизованого модуля для декодування турбо-кодів на основі пошуку максимально правдоподібної бінарної послідовності.

Тип роботи: бакалаврська кваліфікаційна робота (БКР)
(бакалаврська кваліфікаційна робота / магістерська кваліфікаційна робота)

Підрозділ: кафедра автоматизації та інтелектуальних інформаційних технологій, факультет інтелектуальних інформаційних технологій та автоматизації, 1AKIT-216

(кафедра, факультет, навчальна група)

Коефіцієнт подібності текстових запозичень, виявлених у роботі системою StrikePlagiarism 26,4 %

Висновок щодо перевірки кваліфікаційної роботи (відмітити потрібне):

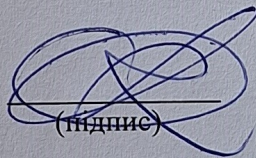
- ☒ Запозичення, виявлені у роботі, є законними і не містять ознак плагіату, фабрикації, фальсифікації. Роботу прийняти до захисту.
- ☐ У роботі не виявлено ознак плагіату, фабрикації, фальсифікації, але надмірна кількість текстових запозичень та/або наявність типових розрахунків не дозволяють прийняти рішення про оригінальність та самостійність її виконання. Роботу направити на доопрацювання.
- ☐ У роботі виявлено ознаки плагіату та/або текстових маніпуляцій як спроб укриття плагіату, фабрикації, фальсифікації, що суперечить вимогам законодавства та нормам академічної доброчесності. Робота до захисту не приймається.

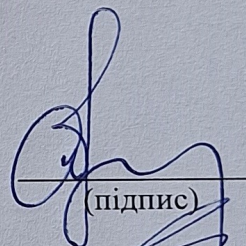
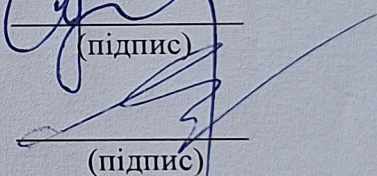
Експертна комісія:

Зав. каф. АПТ, д.т.н., професор Бісікало О. В.
(прізвище, ініціали, посада)

Секретар кафедри АПТ Янчук Л. М.
(прізвище, ініціали, посада)

Особа, відповідальна за перевірку

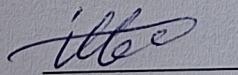

(підпис)


(підпис)

(підпис)

Овчинников К. В.
(прізвище, ініціали)

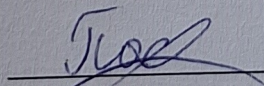
З висновком експертної комісії ознайомлений.

Керівник


(підпис)

доцент Іванов Ю. Ю.
(прізвище, ініціали, посада)

Здобувач


(підпис)

Пасічнюк Т. В.
(прізвище, ініціали)