МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Національний аерокосмічний університет ім. М.Є. Жуковського
«Харківський авіаційний інститут»

Факультет радіоелектроніки, комп'ютерних систем та інфокомунікацій

Кафедра аерокосмічних радіоелектронних систем

Пояснювальна записка до дипломної роботи

(тип кваліфікаційної роботи)

Магістра

(освітній ступінь)

на тему «Дослідження методів завадостійкого кодування в радіоканалах зв'язку з космічними апаратами»

ХАІ. 501.566м.172.1505153 ПЗ

Виконав: студент 6 курсу групи № 566 М
Спеціальність 172 «Телекомунікації та радіотехніка»
(код та найменування)
Освітня програма «Радіоелектронні пристрої, системи та комплекси»
(найменування)
Ільїн І.А.
(прізвище та ініціали студента)
Керівник: Мазуренко О.В.
(прізвище та ініціали)
Рецензент: Биков В.М.
(прізвище та ініціали)

Харків – 2020

Міністерство освіти і науки України
Національний аерокосмічний університет ім. М. Є. Жуковського
«Харківський авіаційний інститут»

Факультет радіоелектроніки, комп'ютерних систем та інфокомунікацій

Кафедра аерокосмічних радіоелектронних систем

Рівень вищої освіти другий (магістерський)

Спеціальність 172 «Телекомунікації та радіотехніка»
(код і найменування)

Освітня програма Радіоелектронні пристрої, системи та комплекси
(найменування)

ЗАТВЕРДЖУЮ
Завідувач кафедри

_____ _____
(підпис) (ініціали та прізвище)
«___» _____20____ р.

З А В Д А Н Н Я
НА КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ

Ільїн Ілля Анатолійович
(прізвище, ім'я та по батькові)

1. Тема кваліфікаційної роботи: «Дослідження методів завадостійкого кодування в радіоканалах зв'язку з космічними апаратами»

керівник кваліфікаційної роботи к.т.н., доцент Мазуренко О. В.
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом Університету №1706-уч від «19» жовтня 2020 року.

2. Термін подання студентом кваліфікаційної роботи 14 грудня 2020 р

3. Вихідні дані до роботи _____

4. Зміст пояснювальної записки (перелік завдань, які потрібно розв'язати)_____

5. Перелік графічного матеріалу _____

6. Консультанти розділів кваліфікаційної роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
1 Рекомендовані методи обробки сигналів в космічних радіолініях зв'язку	**Мазуренко О. В., доцент к. 501**		
2 Математичне моделювання та порівняльний аналіз радіоліній з різними методами завадозахищеного кодування	**Мазуренко О. В., доцент к. 501**		
3 Програмно-апаратна реалізація LDPC і турбо кодування	**Мазуренко О. В., доцент к. 501**		
4 Економічна частина	**Хлівна І. В., професор к. 601**		

Нормоконтроль _____ _____ «____» _____20___ р.
(підпис) (ініціали та прізвище)

7. Дата видачі завдання «19» жовтня 2020 р.

КАЛЕНДАРНИЙ ПЛАН

№ п/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів кваліфікаційної роботи	Примітка
1	**Огляд літературних джерел за темою роботи**	**19.10 – 26.10**	**виконано**
2	**Оформлення основної частини випускної роботи магістра**	**27.10 – 2.11**	**виконано**
3	**Математичне моделювання LDPC і турбо кодування**	**3.11 – 17.11**	**виконано**
4	**Збір та оформлення інформації про програмно-апаратну реалізацію турбо та LDPC кодування**	**18.11 – 25.11**	**виконано**
5	**Розрахунок економічних показників розробки**	**26.11 – 1.12**	**виконано**
6	**Оформлення результатів випускної роботи магістра**	**2.12 – 6.12**	**виконано**

Студент _____ _____
(підпис) (ініціали та прізвище)

Керівник кваліфікаційної роботи _____ _____
(підпис) (ініціали та прізвище)

РЕФЕРАТ

Дипломна робота: 107 с., 73 рис., 11 табл., 51 джерел.

Об’єкт дослідження та аналізу – турбо коди та коди з малою густиною перевірок на парність для завадостійкого кодування в радіолініях зв’язку з космічними апаратами.

Предмет дослідження та аналізу – математичні моделі завадостійкого кодування та декодування кодів з низькою густиною перевірок на парність та турбо-кодів з використанням м’яких алгоритмів декодування min-sum та BCJR, відповідно.

Мета роботи – математичне моделювання, аналіз та порівняння двох видів завадостійкого кодування, розгляд методів спрощення м’яких алгоритмів декодування для програмно-апаратної реалізації на програмованих логічних інтегральних схемах та практичного застосування.

Методи дослідження та аналізу – побудови математичних моделей симуляцій методів завадостійкого кодування в каналах з адитивним білим Гаусовим шумом в програмному пакеті MATLAB.

В роботі розглянуті види космічних місій зв’язку, рекомендовані частотні діапазони та види модуляції для використання в радіолініях зв’язку з космічними апаратами та рекомендовані методи завадостійкого кодування в цих радіолініях. Виділені їх переваги та недоліки. Проведено математичне моделювання, аналіз та порівняння двох методів завадостійкого кодування в радіолініях зв’язку з космічними апаратами за допомогою математичних моделей їх алгоритмів декодування. Наведено шляхи спрощення м’яких алгоритмів декодування цих кодів для практичного застосування та реалізації на програмованих логічних інтегральних схемах.

Результати роботи пропонується використовувати в навчальному процесі та в реальних проектах для реалізації завадостійкого кодування за допомогою турбо-кодів та кодів з низькою густиною перевірок на парність в пристроях зв’язку з космічними апаратами.

РАДІОЛІНІЯ ЗВ’ЯЗКУ З КОСМІЧНИМИ АПАРАТАМИ, МОДУЛЯЦІЯ, ЧАСТОТНИЙ ДІАПАЗОН, ТУРБО-КОДИ, КОДИ З НИЗЬКОЮ ГУСТИНОЮ ПЕРЕВІРОК НА ПАРНІСТЬ, КОДЕР, ДЕКОДЕР, ДЕКОДУВАННЯ, АПОСТЕРІОРНА ІМОВІРНІСТЬ, ЗМІННИЙ ВУЗОЛ, ПЕРЕВІРОЧНИЙ ВУЗОЛ, ІТЕРАТИВНЕ ДЕКОДУВАННЯ, МАТЕМАТИЧНЕ МОДЕЛЮВАННЯ, ПРОГРАМНО-АПАРАТНА РЕАЛІЗАЦІЯ.

THE ABSTRACT

Master's degree work: 107 p., 73 fig., 11 tabl., 51 sources.

Object of research and analysis – turbo codes and low-density parity check codes for noiseless coding in spacecraft radio lines.

Subject of research and analysis – mathematical models of noiseless coding and decoding of low-density parity check codes and turbo codes using min-sum and BCJR algorithms, respectively.

Goal of the work – mathematical modeling, analysis and comparison of two types of noiseless coding, consideration of methods of simplification of soft decoding algorithms for software and hardware implementation on programmable logic integrated circuits and practical application.

Methods of research and analysis – construction of mathematical models of simulations of noiseless coding methods in channels with additive white Gaussian noise in the MATLAB software package.

The paper considers the types of space communication missions, recommended frequency bands and types of modulation for use in radio communication with spacecraft and recommended methods of noiseless coding in these radio lines. Was highlighted their advantages and disadvantages. Mathematical modeling, analysis and comparison of two methods of noiseless coding in radio communication lines with spacecraft using mathematical models of their decoding algorithms. There are ways to simplify the soft decoding algorithms for these codes for practical application and implementation on programmable logic integrated circuits.

The results of the work are proposed to be used in the educational process and in real projects for the implementation of noiseless coding with the help of turbo-codes and codes with a low density of parity check in communication devices with spacecraft.

SPACE COMMUNICATION RADIO LINE, MODULATION, FREQUENCY BAND, TURBO-CODES, CODES WITH LOW-DENSITY OF PARITY CHECKS, ENCODER, DECODER, DECODING, A POSTERIORI PROBABILITY, VARIABLE NODE, CHECK NODE, ITERATIVE DECODING, MATHEMATICAL MODELING, HARDWARE REALISATION.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

AR4JA – Accumulate, Repeat-by-4, and Jagged Accumulate (Накопичування, Повторення 4 рази, та Нерівномірне Накопичування);

BP – Belief Propagation (Розповсюдження Довіри);

BPSK (ДФМ) – Binary Phase-Shift Keying (Двійкова Фазова Модуляція);

LDPC – Low Density Parity Check (Низька Щільність Перевірок на Парність);

LLR (ЛКП) – Logarithmic Likelihood Ratio (Логарифмічний Коефіцієнт Правдоподібності);

PSK (ФМ) – Phase-Shift Keying (Фазова Модуляція);

QPSK (КФМ) – Quadrature Phase-Shift Keying (Квадратурна Фазова Модуляція);

SOVA – Soft Output Viterbi Algorithm (Алгоритм Вітербі з М'яким Виходом);

SPA – Sum-Product Algorithm (Алгоритм Суми Добутків);

A_{Σ} – сумарне ослаблення сигналу на дільниці між антенами, дБ;

$A_{\text{прд}}$ – ослаблення сигналу у фільтрах між антенами і виходами передавача, дБ;

$A_{\text{прм}}$ – ослаблення сигналу у фільтрах між антенами та входом приймача, дБ;

$A_i(t)$ – дискретне значення амплітуди, В;

A_{MN} – матриця циркулянт розмірністю $Q \times Q$;

G LDPC – генераторна матриця розмірністю $X \times Y$;

H LDPC – перевірочна матриця розмірністю $X \times Y$;

a_{∂} – погонне ослаблення сигналу, дБ/м;

b^c – доповнення до b;

d_v – максимальний ступінь змінного вузла;

d – відстань Хеммінга;

d_{free} – вільна відстань коду;

H – висота супутника над поверхнею Землі, м;

$I_{X \to Y}$ – зовнішня інформація, що надсилається від солдата X солдату Y;

$I_{Z \to X}$ – зовнішня інформація, що надсилається від солдата Z солдату X;

$I_{Y \to X}$ – зовнішня інформація, що надсилається від солдата Y солдату X;

I_X – внутрішня інформація;

$K_{\text{пол}}$ – величина поляризаційних втрат;

k – стала Больцмана;

k – довжина інформаційного слова, бітів.

L – довжина лінії зв'язку між двома ЗС, м;

L_j – значення КП, обчислене на основі вибірки y_j з каналу;

$L_{sys}(\hat{u}_i)$ – систематична складова надійності;

$L_{app}(\hat{u}_i)$ – апріорна складова надійності;

$L_{ext}(\hat{u}_i)$ – зовнішня складова надійності;

M_i – кількість точок на окружності радіусом $A_i(t)$;

$N(X)$ – сукупність сусідів солдата X;

$P_{\text{твх}}$ – потужність власних шумів приймача, Вт;

$P_{\text{ф}}$ – потужність шумів фідеру, віднесена до входу приймача, Вт;

P_a – потужність шумів антени, віднесена до входу приймача, Вт;

$P_{\text{к}}$ – потужність космічних шумів, Вт;

$\Pr(u_i = x)$ – апріорна імовірність;

R_∂ – протяжність траси, на якій спостерігаються опади, м;

$T_{e\text{з}}$ – еквівалентна температура Землі, віднесена до входу антени, K;

T_{ea} – еквівалентна температура атмосфери, віднесена до входу антени, K;

$\hat{u}$ – прийняте рішення про декодоване слово;

$\hat{u}_i$ – прийняте рішення про декодований біт;

$V^2(t)$ – множник ослаблення;

$\hat{v}$ – орієнтовне декодоване кодове слово;

v – кодова послідовність довжини k;

v' – кодова послідовність довжини k;

w_{free} – сумарна вага інформаційних слів, яким відповідають кодові слова з вагою d_{free};

η – коефіцієнт корисної дії фільтрів і фідерів між входом антени і приймачем;

λ_d – частка всіх ребер, приєднаних до змінного вузла ступеня d_v;

Π_e – еквівалентна смуга пропуску приймача, Гц;

ρ_d – частка всіх ребер, приєднаних до перевірочного вузла ступеня d_v;

σ – дисперсія шуму;

$\varphi_j(t)$ – дискретне значення фазового кута, рад;

ϕ_0 – початкова фаза несучого сигналу, рад;

$\Omega_{\text{з}}$ – тілесний кут Землі, що спостерігається з борта супутника, стерадіан;

ω_0 – кутова частота несучого коливання сигналу, рад/с;

$\|\cdot\|^2$ – Евклідова відстань;

ЗС – земна станція;

КП – коефіцієнт правдоподібності;

ССЗ – система супутникового зв’язку;

ССМ – супутникова служба мовлення;

ШСЗ – штучний супутник Землі;

ВСТУП

В даний час спостерігається бурхливий розвиток та зростання систем передачі інформації даних з лавиноподібним збільшенням об’єму інформації, що передається та приймається. Розвиток супутникового мовлення та телефонії, наземного телевізійного та мобільного мовлення, кабельних, локальної і міської мережі та іншого неможливе без розвитку засобів завадостійкого кодування на всіх рівнях. Швидкість та об’єм передачі даних через вищезазначені радіоелектронні мережі зростає майже за експоненціальною залежністю, як і зашумленість частотного діапазону, що призводить до нових більш жорсткіших вимог до надійності передачі інформації. Одним з головних методів захисту інформації, що передається по каналу, від спотворень та втрат є завадостійке кодування. Кодери та декодери – це найважливіша частина архітектури побудови будь-яких ліній цифрової передачі даних з особливими вимогами до надійності виправлення помилок.

Під час проектування радіотехнічної системи, надійно захищеної від перешкод, перед розробниками стоїть проблема вибору завадостійкого кодека, при цьому необхідно враховувати безліч обмежень на системному та апаратному рівнях та вимог стандартів передачі даних.

Корегуючі декодери на основі кодів з малою густиною перевірок на парність (англ. – Low Density Parity Check Codes – LDPC codes) стали в останнє десятиліття популярними в системах зв’язку, завдяки їх високій продуктивності та можливості паралельної апаратної реалізації. Програмовані логічні інтегральні схеми ідеально підходять для малосерійного випуску LDPC-декодерів завдяки можливості їх перепрограмування.

Турбо-коди також з упевненістю можна назвати одними з найбільш затребуваних в сучасних мережах. Вони довели свою ефективність в стандартах WiMax та DVB-RCS. Але найбільшого поширення ці коди отримали в мобільних мережах третього та четвертого поколінь UMTS та 3GPP LTE. Їх широке практичне застосування обумовлене ефективним ітеративним турбо-декодером, і можливістю використання одного декодера для різних довжин інформаційних слів та гнучкої схеми регулювання швидкості коду.

Отже, метою роботи є аналіз та порівняння найефективніших методів завадостійкого кодування, які рекомендовані для використання в радіолініях зв’язку з космічними апаратами при різних параметрах обраних кодів, та з урахуванням і описом властивостей кожного.

1 РЕКОМЕНДОВАНІ МЕТОДИ ОБРОБКИ СИГНАЛІВ В КОСМІЧНИХ РАДІОЛІНІЯХ ЗВ'ЯЗКУ

Система радіозв'язку з космічними апаратами призначена для передачі інформації на велику відстань за допомогою радіосигналів.

Інформація, висловлена у визначеній формі являє собою повідомлення, що підлягає передачі на велику відстань. Для надання інформації використовується мова, яка характеризується сукупністю знаків і правилами їх застосування. Сукупність знаків, що містять інформацію і є повідомленням. Повідомлення може бути безперервним(аналоговим) і дискретним.

Для передачі інформації в системі радіозв'язку з космічними апаратами, повідомлення необхідно перетворити у первинний електричний сигнал.

Перетворення дискретних первинних електричних сигналів у комбінації елементарних сигналів називається кодуванням.

Первинний електричних сигнал є, як правило, низькочастотним і його неможливо випромінювати в середовищі поширення радіохвиль. Тому він повинен бути перетворений у високочастотний сигнал, який називається радіосигналом.

Перетворення первинного електричного сигналу в радіосигнал здійснюється шляхом зміни одного чи декількох параметрів несучої радіочастоти. Процес зміни одного чи декількох параметрів несучої частоти відповідно до змін параметрів переданого первинного електричного сигналу називається модуляцією. Якщо модуляція здійснюється дискретними сигналами, то її, як правило, називають маніпуляцією.

Таким чином, при передачі повідомлень на передавальній стороні здійснюється сукупність операцій: первинне перетворення, кодування, модуляція, посилення і випромінення.

На стороні приймача здійснюються зворотні операції: прийом радіохвиль, посилення і фільтрація високочастотних коливань, демодуляція, декодування й перетворення сигналу у повідомлення.

Джерело й одержувач повідомлення, технічні пристрої, що забезпечують передачу повідомлень, а також середовище поширення радіохвиль складають систему радіозв'язку. Саме середовище поширення радіохвиль між передавальною і приймальною антенами називається лінією радіозв'язку.

Лінія радіозв'язку і сукупність технічних засобів, що забезпечують передачу сигналів від джерела повідомлення до одержувача, називається каналом радіозв'язку. Структурна схема системи радіозв'язку має вигляд, показаний на рисунку 1.1.

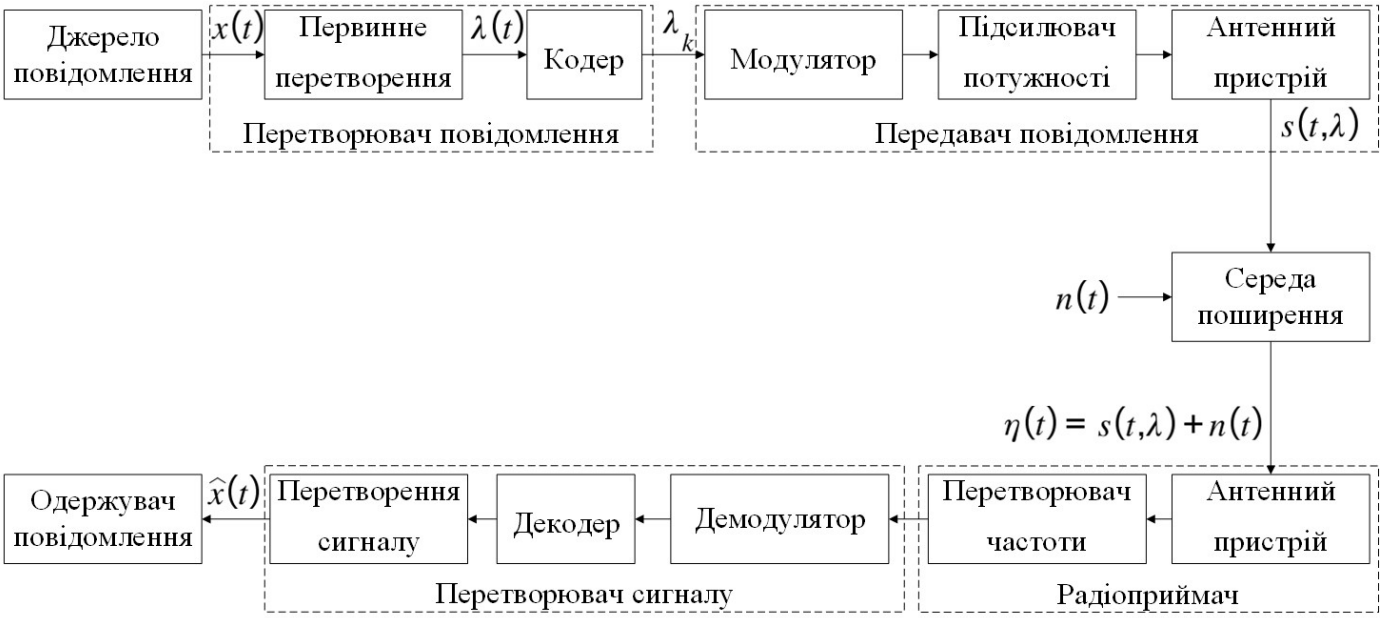

Рисунок 1.1 – Структурна схема системи радіозв’язку

Характерною рисою системи радіозв’язку з космічними апаратами є спотворення сигналів за рахунок перешкод. Внаслідок цього до одержувача надходить повідомлення, яке у загальному випадку відрізняється від переданого і є лише його оцінкою[1].

Отже, в цьому розділі будуть розглянуті перспективні методи боротьби зі спотворенням сигналів із застосуванням завадостійкого кодування, види місій в радіолініях зв’язку, частотні діапазони, використовувані для зв’язку з космічними апаратами, а також методи модуляції рекомендовані для таких радіоліній зв’язку.

1.1 Види місій в радіолініях зв’язку з космічними апаратами

Система супутникового зв’язку (ССЗ) включає в себе космічну станцію (супутник) і сукупність наземних станцій. Земна станція (ЗС – радіостанція космічної служби радіозв’язку, яка знаходиться або на поверхні Землі, або в основній частині земної атмосфери.

Згідно регламенту систем зв’язку всі ССЗ використовуються в складі таких служб радіозв’язку:

— фіксованої супутникової служби (зв’язок між ЗС, розташованими у фіксованих пунктах);

— рухомої супутникової служби (зв’язок між рухомими ЗС за допомогою одного або декількох супутників зв’язку);

— супутникової служби мовлення (ССМ) (передача телевізійних або звукових програм одному або групі абонентів без використання проміжних технічних засобів).

За територією обхвату, приналежності і призначенню всі ССЗ і ССМ поділяються на:

— міжнародні (Intelsat);

— національні (Екран, Eutelsat, Arabsat);

— відомчі (Кристал, Флітсатком).

В останні роки частіше використовується розподіл на глобальні (Iridium, Globalstar, ICO, Inmarsat) і регіональні (Thuraya, AceS, Зеркало-К).

У залежності від наявності на борту штучного супутника Землі (ШСЗ) радіоапаратури для посилення сигналів, які ретранслюються, розрізняють ССЗ:

— з пасивним ШСЗ;

— з активним ШСЗ;

— з негайною ретрансляцією сигналу;

— із затриманою ретрансляцією сигналу.

У свою чергу супутники зв'язку, які є головною складовою частиною ССЗ, можна поділити на спеціалізовані і багатофункціональні. Спеціалізовані супутники використовуються для рішення однієї задачі (передача телефонних або телевізійних повідомлень, роботи в складі ССМ).

Один багатофункціональний супутник може працювати в складі декількох ССЗ і в той же час у складі однієї ССЗ може використовуватися декілька ШСЗ.

У залежності від висоти польоту ШСЗ або іншого космічного апарата космічні місії діляться на категорії:

— категорія А – висота польоту космічного апарату над Землею менше ніж 2 мільйони кілометрів;

— категорія Б – висота польоту космічного апарату над Землею більше або дорівнює 2 мільйонам кілометрів.

Також місії в радіолініях зв'язку з космічними апаратами поділяються за призначенням так наведені в таблиці 1.1.

Таблиця 1.1 – Розподіл місій за призначенням

Призначення	Супутникова система
Цивільний далекий зв'язок: Міжнародний Регіональний	Intelsat, «Интерспутник», Anik, Satcom, Westar, «Молния»
Віщальний зв'язок	ATS-6, CTS, OTS
Зв'язок з військовими об'єктами	DSCS-III, Skynet, NATO
Зв'язок з морськими об'єктами: Міжнародний Регіональний	Inmarsat, Marisat, Marots
Зв'язок з літальними апаратами	Aerosat
Навігаційний зв'язок з морськими і повітряними об'єктами	Navstar, GPS
Дослідження космічного простору	TDRSS, Meteosat, «Космос»
Метеорологічний зв'язок	SMS/60ES, 6MS-1
Дослідження Землі	Landsat, Seasat

На сьогодні основними напрямками розвитку ССЗ є:

— використання широкосмугових супутникових технологій зв’язку та сучасних методів завадостійкого кодування інформації, які забезпечують високошвидкісну та захищену від помилок передачу даних;

— освоєння нових діапазонів радіочастот для забезпечення роботи широкосмугових ССЗ з геостаціонарними і низькоорбітальними супутниками;

— значне розширення спектра послуг зв’язку для кінцевих користувачів: мобільний персональний зв’язок, доступ до інтернету, передача відеоінформації, відеоконференцзв’язок, мультимедійне широкомовлення, послуги визначення місцезнаходження, тощо;

— впровадження нових бортових систем зв’язку, супутникових антен, оптичних систем міжсупутникового зв’язку, спостережних антен абонентських станцій, портативних мобільних терміналів;

— застосування стандартних транспортних протоколів, які адаптовані до фізичних супутникових каналів.

1.2 Основні параметри та частотні діапазони радіоліній зв’язку з космічними апаратами

До основних параметрів ССЗ відносяться:

Сигнал на виході приймальних пристроїв. Розглянемо канал зв’язку протяжністю R, що містить передавач, приймач і антено-фідерні тракти, які характеризуються коефіцієнтами посилення антени $G_{\text{прм}}$ і $G_{\text{прд}}$ та коефіцієнтами корисної дії фідерів $\eta_{\text{прм}}$ і $\eta_{\text{прд}}$. Тоді помилковість сигналу на вході приймача можна записати як[2]:

$$P_{\text{свх}} = \frac{P_{\text{прд}} G_{\text{прд}} G_{\text{прм}} \eta_{\text{прд}} \eta_{\text{прм}} V^2(t)}{A_{\Sigma} A_{\text{прд}} A_{\text{прм}}} K_{\text{пол}}, \tag{1.1}$$

де A_{Σ} – сумарне ослаблення сигналу на дільниці між антенами, дБ;

$A_{\text{прд}}$ – ослаблення сигналу у фільтрах між антенами і виходами передавача, дБ;

$A_{\text{прм}}$ – ослаблення сигналу у фільтрах між антенами і входом приймача, дБ;

$V^2(t)$ – множник ослаблення, що не перевищує поріг протягом t часу;

$K_{\text{пол}}$ – величина поляризаційних втрат.

Добуток $P_{\text{прд}} G_{\text{прд}} G_{\text{прм}}$ називається ефективною потужністю ізотропного випромінювача. A_{Σ} визначається ослабленням сигналу у вільному просторі A_0 і поглинанням у атмосфері при куті узвишшя β в разі відсутності опадів $A_a(\beta)$.

$$A_{\Sigma} = A_0 A_a(\beta) = \left(\frac{4\pi r}{\lambda}\right)^2 A_a(\beta), \tag{1.2}$$

$$a_a = 10 lg A_a(\beta). \tag{1.3}$$

Множник ослаблення $V^2(t)$ визначається тільки поглинанням електромагнітної енергії в опадах (дощ, хмари, тумани).

$$V^2(t) = 10^{-0.1 a_\partial R_\partial \cdot 10^3}, \quad (1.4)$$

де a_∂ – погонне ослаблення сигналу, дБ/м ;

R_∂ – протяжність траси, на якій спостерігаються опади, м.

Величина a_∂ для опадів різної інтенсивності визначається за графіками. Для вертикальної траси $R_\partial = 3 \dots 4$ м, а для горизонтальної залежить від інтенсивності опадів:

$$I < 10\text{мм/г} - \ R_\partial = n \cdot 100 \text{ м};$$

$$I = 10\text{мм/г} - \ R_\partial = 45 \dots 55 \text{ м};$$

$$I = 25 \dots 30\text{мм/г} - \ R_\partial = 30 \dots 35 \text{ м};$$

$$I > 100\text{мм/г} - \ R_\partial = 8 \dots 12 \text{ м}.$$

Відсоток часу, протягом якого можуть спостерігатися опади визначається по спеціальних кривих. При використанні антен з однаковою поляризацією $K_{\text{пол}} = 1$. Якщо одна з антен має кругову поляризацію, а інша лінійну, то $K_{\text{пол}} = 0.5$.

Шуми на вході приймальних пристроїв. У супутникових системах зв'язку використовуються приймачі з як можна нижчим рівнем власних шумів.

Сумарна потужність шумів, віднесена до входу приймача:

$$P_{\text{ш}\Sigma} = P_{\text{твх}} + P_{\text{ф}} + P_a \eta + P_{\text{к}} \eta, \quad (1.5)$$

де $P_{\text{твх}}$ – потужність власних шумів приймача, Вт;

$P_{\text{ф}}$ – потужність шумів фідеру, віднесена до входу приймача, Вт;

P_a – потужність шумів антени, віднесена до входу приймача, Вт;

$P_{\text{к}}$ – потужність космічних шумів, Вт;

η – коефіцієнт корисної дії фільтрів і фідерів між входом антени і приймачем.

Враховуючи, що потужність шумів пов'язана з еквівалентною шумовою температурою T_e залежністю:

$$P_{\text{ш}} = kT_e \text{П}_e, \quad (1.6)$$

де k – стала Больцмана;

П_e – еквівалентна смуга пропуску приймача.

$$P_{\text{ш}\Sigma} = T_{\text{епр}} + T_{\text{еф}} + (T_{ea} + T_{ee})\eta, \quad (1.7)$$

Власні шуми приймача, віднесені до його входу, прийнято характеризувати коефіцієнтом шуму Ш або еквівалентною шумовою температурою. Ці параметри пов'язані співвідношенням:

$$T_{\text{епр}} = T_0(\text{Ш} - 1), T_0 = 290K \tag{1.8}$$

$T_{\text{еф}} = T_{\text{ф}}(1 - \eta)$ (згасання 0.1 дБ $\rightarrow \eta = 0.977 \rightarrow T_{\text{еф}} = 6.7\ K$).

Отже, доцільно вхідні малошумові підсилювачі приймача встановлювати безпосередньо поблизу випромінювачів антени.

$$T_{ea} = T_{ез} + T_{ea}, \tag{1.9}$$

де $T_{ез}$ – еквівалентна температура Землі, віднесена до входу антени, K;

T_{ea} – еквівалентні температури атмосфери, віднесена до входу антени, K.

$$T_{ез} = 23 + 0.2(90^{\circ} - \beta^{\circ}) \tag{1.10}$$

$$T_{ea} = T_{ea}(\beta) + 23 + 0.2(90^{\circ} - \beta^{\circ}) \tag{1.11}$$

Складова виразу (1.11) $T_{ea}(\beta)$ визначається за допомогою відповідних графіків.

$$T_{ea} = [290 + T_{ea}(90)]\frac{\Omega_з}{\Omega_a}, \tag{1.12}$$

де $\Omega_з$ – тілесний кут Землі, що спостерігається з борта супутника, стерадіан;

Ω_a – тілесний кут головної пелюстки діаграми спрямованості бортової антени, стерадіан.

$T_{\text{ек}}$ визначається із відповідних графіків у всіх випадках, коли приймальна антена не направлена на сонце, місяць і дискретні космічні джерела.

Перед тим як розглядати частотні діапазони, які використовуються в радіолініях зв'язку з космічними апаратами слід розглянути деякі особливості передачі сигналів у ССЗ. Першою такою особливістю є запізнення сигналу.

Велика довжина лінії зв'язку між земними станціями і ретранслятором, що знаходиться на борту ШСЗ, призводить до запізнення сигналів:

$$\Delta t = \frac{L \cdot 10^3}{c} \approx \frac{2H}{c}, \tag{1.13}$$

де L – довжина лінії зв'язку між двома ЗС, м;

H – висота супутника над поверхнею Землі, м.

Для геостаціонарних супутників ($H = 3600000$ м) $\Delta t \approx 250 \cdot 10^{-3}$ с. Це веде до появи вимушених пауз при дуплексних телефонних розмовах.

Іншим явищем, яке слід згадати є луна-сигнали [1].

Запізнення сигналів призводить до появи помітних для абонентів луна-сигналів, виникаючих при переході з чотирьох провідних ланцюгів зв'язку на дводротові через неідеальність диференціальних систем. Наявність луна-сигналів призводить до прослуховування абонентом своєї власної розмови,

затриманої на час, який дорівнює подвоєному часу поширення сигналу між абонентами:

$$t_{\text{луна}} \approx \frac{4H}{c}. \tag{1.14}$$

У цих випадках необхідно забезпечити згасання луна-сигналів до 60 дБ відносно рівня корисного сигналу.

І ще одна особливість, яку не можна забувати – це ефект Доплера.

Відомо, що при русі джерела сигналу з швидкістю v_r частота коливань f, що приймаються, пов'язана з частотою f_0 коливань, що випромінюються, співвідношенням [3]:

$$f = \frac{f_0}{\left(1 \pm \frac{v_r}{c}\right)}. \tag{1.15}$$

Знак + має місце при збільшенні відстані між випромінювачем і приймачем.

Враховуючи, що $v_r/c \ll 1$ – зміна частоти, викликана ефектом Доплера, дорівнює

$$\delta f = f - f_0 = \pm f_0 \frac{v_r}{c}. \tag{1.16}$$

Для ССЗ під v_r розуміємо радіальну складову вектору швидкості супутника ретранслятора (співпадаючу з лінією радіозв'язку ШСЗ – ЗС).

Найбільш сильніше ефект Доплера виявляється в ССЗ з еліптичними орбітами. На робочій ділянці орбіти $v_r/c \leq 10^{-5}$. У діапазоні 8/7 ГГц доплерівський зсув частоти складає $f = 8 \cdot 10^9 \cdot 10^{-5} = 80 \cdot 10^3$ Гц.

Ефект Доплера призводить не тільки до зміни несучої частоти, але і викликає деформацію спектра повідомлення, що передається. Так, якщо модуляція здійснювалася коливанням з частотою F, прийняте коливання на виході детектора з урахуванням ефекту Доплера буде мати частоту.

$$F' = F\left(1 \pm \frac{v_r}{c}\right). \tag{1.17}$$

Тому при модуляції коливаннями з частотами $F_1 = 10^3$ Гц і $F_2 = 10^7$ Гц, на виході детектора, при $\frac{v_r}{c} = 10^{-5}$, отримаємо відповідні частоти 10^3 Гц $\pm 10^{-2}$ Гц і 10 МГц $\pm$ 100 Гц.

Звідси випливає, по-перше, що верхні частоти в спектрі повідомлення будуть змінюватися на велику величину, а по-друге, що ширина спектру сигналу, який приймається, буде відрізнятися від ширини спектра коливань, які модулюють.

Таким чином радіолінії зв'язку з космічними апаратами є складними радіотехнічними системами, при дослідженні і проектуванні яких необхідно враховувати велике число різних чинників, що впливають на її ефективність. Поряд з цим ССЗ забезпечує рішення ряду завдань, що не вирішуються іншими системами зв'язку, а саме:

— стійкість глобального зв'язку;

— високу пропускну спроможність каналу зв'язку;

— можливість багатоканального зв’язку і передачі широкосмугових сигналів;

— високу скритність і завадостійкість повідомлень, що передаються;

— можливість сполучення з іншими перспективними системами і мережами інформаційного зв’язку та обміну інформацією (системи персонального мобільного зв’язку, мереж доступу до інтернету, мультимедійні системи, тощо).

Після розгляду параметрів та особливостей радіолінії зв’язку з космічними апаратами, перейдемо до частотних діапазонів, використовуваних в радіолінії.

Отже, вибір робочих частот для радіолінії зв’язку з космічними апаратами визначається наступними факторами:

— рівнем зовнішніх джерел шумів, які приймають антени;

— умовами поширення та поглинання радіохвиль;

— наявністю відповідних технічних засобів;

— взаємними завадами між ССЗ та іншими службами.

— дальністю між передавачем та приймачем;

— потужністю передавача;

— розмірами бортової та наземної антени;

— довжиною хвилі;

— якістю приймально-передавальної електроніки;

— швидкістю руху космічного апарату.

Першим двом вимогам найбільшою мірою відповідає діапазон частот 1 ... 10Гц, оскільки на більш високих частотах різко збільшується рівень шумів та поглинання в кисні і водяних парах.

У діапазоні частот до 27.5 ГГц для різних служб супутникового зв’язку, у залежності від району, відповідно до регламенту супутникової служби зв’язку, виділені спеціальні смуги частот. Виділені смуги частот або групи смуг частот називають і означають за округленими значеннями частот на дільницях Земля – Супутник і Супутник – Земля. Широко використовують поняття діапазонів 6/4ГГц, 8/7ГГц, 14/11ГГц, 30/20ГГц.

Ширина смуги частот, яка може бути виділена для окремого ретранслятора в кожному діапазоні, обмежена значенням 500 МГц у смугах 6/4, 8/7, 14/11 і до 3.5ГГц в діапазоні 30/20ГГц.

Для військових систем зв’язку виділені смуги частот: Земля – ШСЗ – 7.90…8.40ГГц, а ШСЗ – Земля – 7.250…7.750ГГц.

У авіаційно-космічних системах радіозв’язку, призначених для зв’язку з літальними апаратами, застосовується більш низькочастотний діапазон 150…400МГц. Рівні шумів і втрати при поширенні для літакової антени з діаграмою спрямованості у вигляді півсфери в цьому діапазоні є цілком допустимими і приблизно постійними.

Таким чином в таблиці 1.2 приведені всі частотні діапазони, які використовуються в радіолініях зв’язку з космічними апаратами [4].

Таблиця 1.2 – Діапазони радіоліній космічного зв’язку та їх позначення

Діапазон, ГГц	Робочі частоти, ГГц	Проблеми освоєння діапазону	Типове застосування
P 0.230 – 1.0	Різні смуги	Навантаження радіоспектра	Супутникова навігація
L 1.530 – 2.7	Різні смуги	Втручання інших систем	Супутникова навігація, голосовий зв’язок
S 2.7 – 3.5	Різні смуги		Супутникова навігація
C 3.7 – 6.5	3.7 – 4.2 (униз) 5.925 – 6.425 (угору)	Втручання інших систем	Фіксований зв’язок, передача відео
X 7.5 – 8.5	7.250 – 7.745 7.900 – 8.395	Втручання інших систем	
Ku(Європа) 11.0 – 14.0	Фіксований зв’язок: 10.7 – 11.7 14.0 – 14.8 Мовлення: 11.7 – 12.5 17.3 – 18.1 Телекомунікація: 12.5 – 12.75 14.0 – 14.8	Компенсування дощових і температурних впливів	Доступ до інтернету, ТВ, мобільний зв’язок, широкосмуговий зв’язок та передача даних
Ku(США) 11.0 – 18.0	Фіксований зв’язок: 11.7 – 12.2 14.0 – 14.5 Мовлення: 12.2 – 12.7 17.3 – 17.8		Доступ до інтернету, ТВ, мобільний зв’язок, широкосмуговий зв’язок та передача даних
Ka 18.0 – 31.0	17.7 – 21.7 27.5 – 30.5	Висока складність і вартість	Високошвидкісна передача даних, мультимедія
V 31.0 – 70.0	На етапі досліджень	Великий обсяг науково-дослідницьких та дослідно-конструкторських робіт	Супутниковий радіозв’язок, радіоастрономія, радіолокація (керування зброєю)

1.3 Методи модуляції рекомендовані для радіоліній зв'язку з космічними апаратами

1.3.1 Модуляція BPSK, QPSK та 8-PSK

Фазова модуляція (англ. – Phase-Shift Keying – PSK) передбачає зміну фази несучого сигналу в залежності від символу, який передається[5]. Кожному із символів модуляції відповідає одне значення фази несучого сигналу. Найпростішим прикладом такої модуляції є модуляція BPSK (англ. – Binary Phase-Shift Keying – BPSK) – двійкова фазова модуляція. Даний вид модуляції використовується достатньо широко завдяки високій завадостійкості та простоті реалізації модулятора та демодулятора.

В системі DVB-S2X модуляція BPSK в поєднанні з завадостійким кодуванням з великою надмірністю використовується для забезпечення завадостійкого прийому сигналу в умовах дуже низьких відношень сигнал/шум.

При BPSK модуляції кожен символ відповідає одному біту початкового інформаційного потоку, тобто швидкість передачі символів дорівнює швидкості передачі біт [5], [6]. $B_r = S_r$. Розглянемо цифровий сигнал $b(t)$ у вигляді послідовності імпульсів (біт), як показано на рисунку 1.2. Сигнал $b(t)$ на рисунку 1.2 називається уніполярним, в ньому інформаційному логічному нулю відповідає значенням сигналу $b(t) = 0$, а сигнал $b_0(t)$ – називається біполярним, в якому інформаційному логічному нулю відповідає значення сигналу $b_0(t) = -1$.

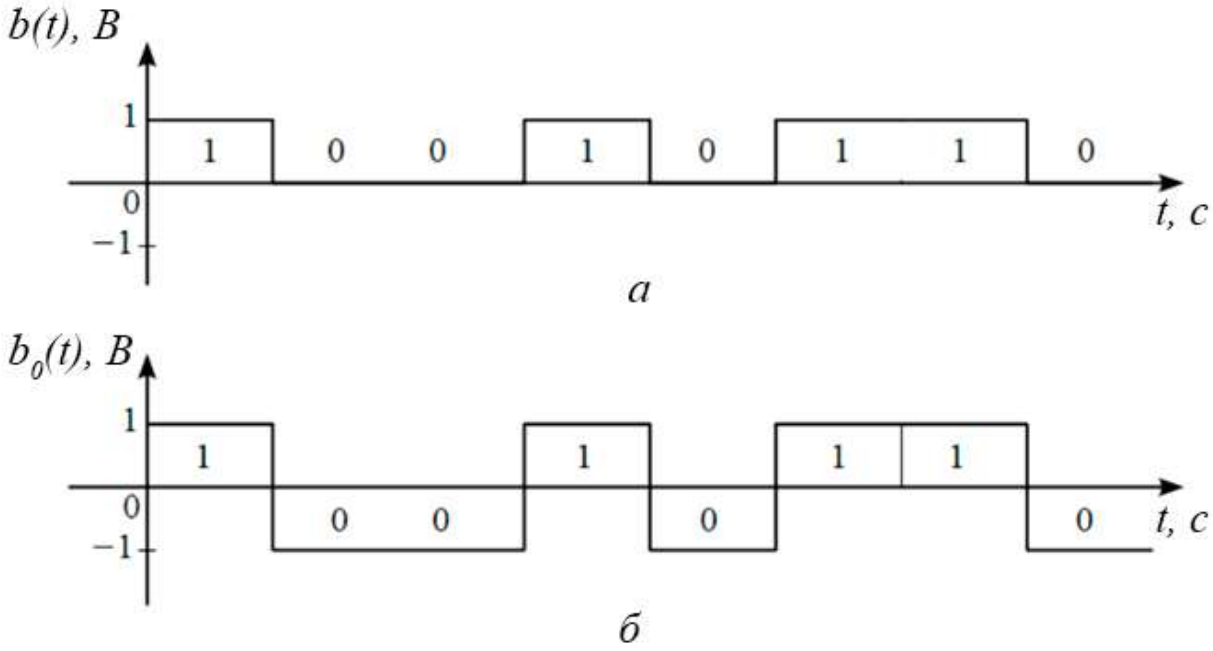

Рисунок 1.2 – Цифровий сигнал:
а – уніполярний;
б – біполярний

Схема фазового модулятора показана на рисунку 1.3. В якості модулюючого сигналу на вході модулятора виступає сигнал $b(t)$. Оскільки даний сигнал приймає значення 0 та 1, синфазна $I(t)$ та квадратурна $Q(t)$ складові комплексної огинаючої сигналу дорівнюють:

$$I(t) = \cos\big(\pi \cdot b(t)\big) = \pm 1 = b_0(t), \tag{1.18}$$

$$Q(t) = \sin\big(\pi \cdot b(t)\big) = 0. \tag{1.19}$$

Девіація частоти сигналу дорівнює π радіан.

Тоді сигнал BPSK можна записати у такому вигляді:

$$S_{bpsk}(t) = I(t) \cdot \cos(\omega_0 \cdot t + \phi_0) + Q(t) \cdot \sin(\omega_0 \cdot t + \phi_0) = \\ = b_0(t) \cdot \cos(\omega_0 \cdot t + \phi_0), \tag{1.20}$$

де ω_0 – кутова частота несучого коливання сигналу, рад/с;

ϕ_0 – початкова фаза несучого сигналу, рад.

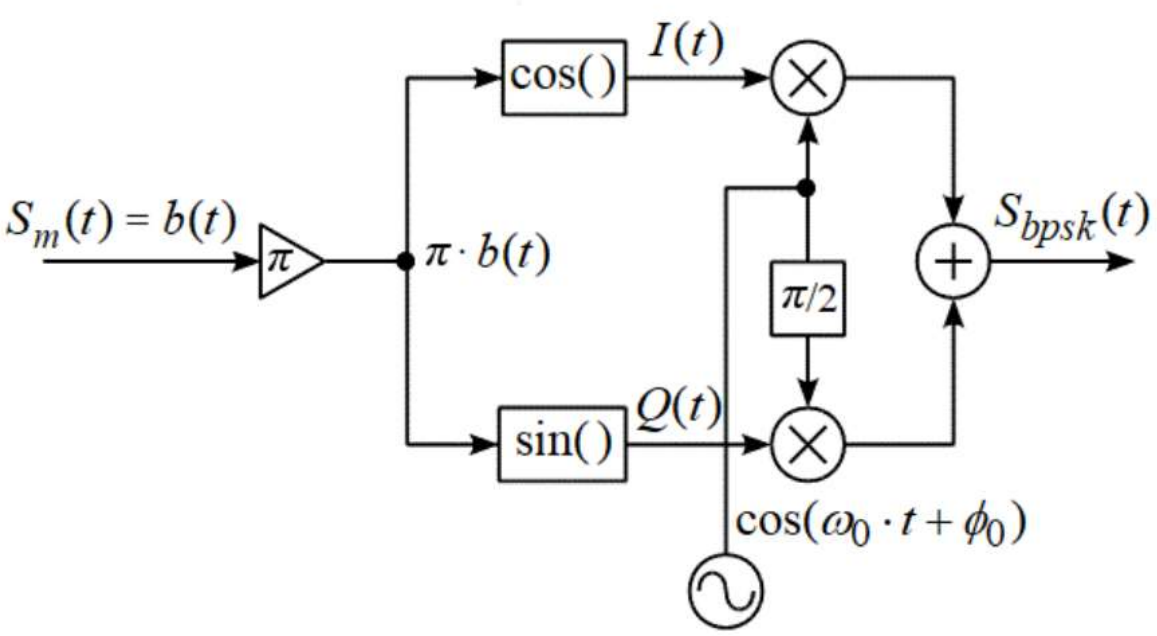

Рисунок 1.3 – Формування BPSK сигналу

Часова діаграма сигналу на виході модулятора представлена на рисунку 1.4. Інформація передається зі швидкістю B_r, біт/с. Тривалість одного імпульсу цифрової інформації дорівнює $T = 1/B_r$. Початковий модулюючий сигнал множиться на несуче коливання $f(t) = \cos(\omega_0 \cdot t + \phi_0)$, в результаті чого отримуємо фазоманіпульований сигнал із стрибком фази на π радіан.

Спектр сигналу BPSK представляє собою перенесений на несучу частоту спектр цифрового біполярного сигналу $b_0(t)$[6]. На рисунку 1.5 показано спектр сигналу BPSK при швидкості передачі інформації $B_r = 20$ кбіт/с та несучій частоті $f_0 = 250$ кГц.

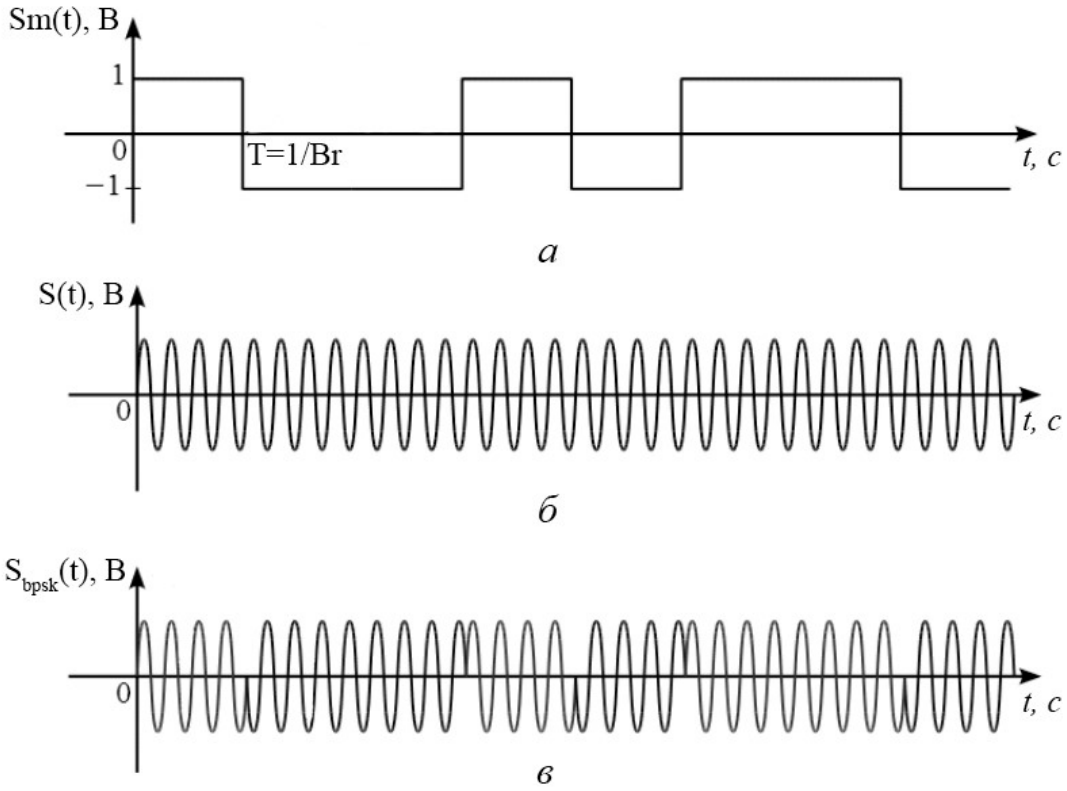

Рисунок 1.4 – Часові діаграми сигналів:
а – цифровий біполярний сигнал;
б – несуче коливання;
в – фазоманіпульований сигнал на виході модулятора

Спектр має основну пелюстку, яка має ширину, чисельно дорівнюючу подвоєній швидкості передачі інформації $2 \cdot B_r$, та бокові пелюстки, які повільно убувають, ширина котрих чисельно дорівнює швидкості B_r.

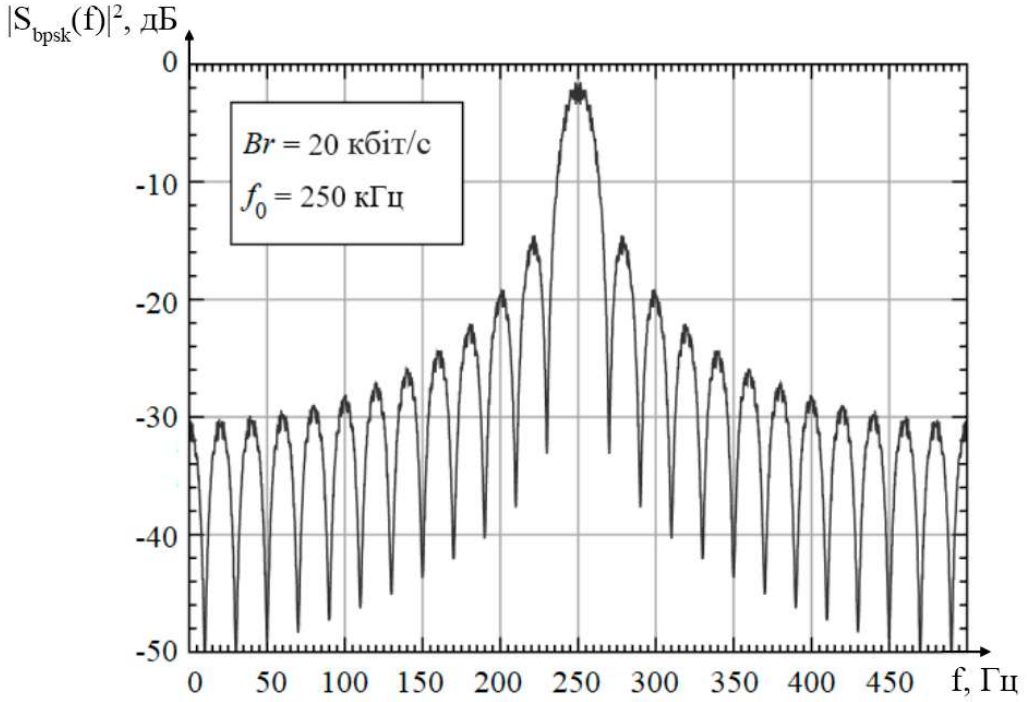

Рисунок 1.5 – Спектр BPSK сигналу

Сузір’я сигнальних точок BPSK модуляції представляє собою дві сигнальні точки, які розташовані на синфазній осі, та мають значення синфазної компоненти -1 та 1. Вигляд сузір’я показано на рисунку 1.6.

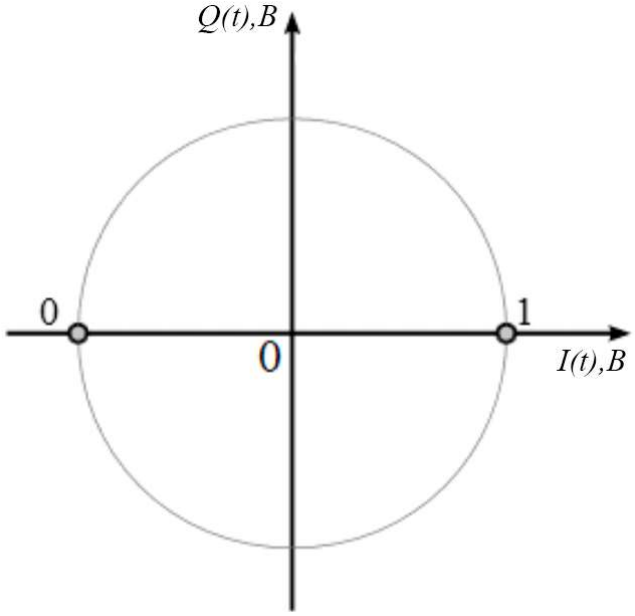

Рисунок 1.6 – Сузір’я сигнальних точок BPSK модуляції

При QPSK (англ. – Quadrature Phase-Shift Keying – QPSK) модуляції – квадратурній фазовій модуляції – один символ модуляції несе в собі два біти початкового інформаційного потоку[5], [6]. При цьому символьна швидкість дорівнює половині швидкості бітового інформаційного потоку: $S_r = B_r/2$. На відміну від BPSK модуляції, при використанні котрої квадратурна компонента $Q(t)$ завжди дорівнює нулю, в QPSK модуляції як синфазна, так і квадратурна компоненти приймають різні значення, які відповідають інформаційним бітам, що передаються. На рисунку 1.7 показано сузір’я сигнальних точок QPSK модуляції.

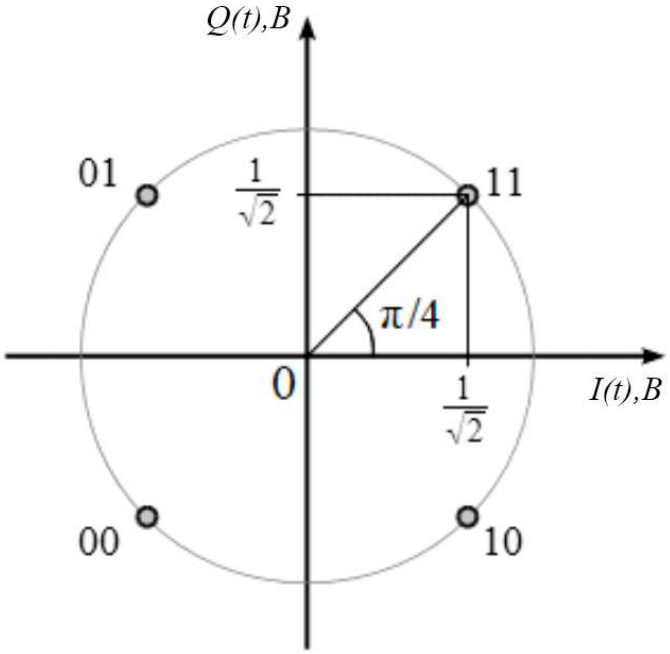

Рисунок 1.7 – Сузір’я сигнальних точок QPSK модуляції

Сузір'я QPSK складається з чотирьох точок, які характеризуються відповідним значенням фазового кута, та які розташовані на одній окружності. Початковий цифровий потік розбивається на парні та непарні біти, в синфазному каналі кодуються парні біти, в квадратурному каналі – непарні. Два послідовно слідуючих біти інформації кодуються одночасно синфазною $I(t)$ та квадратурною $Q(t)$ компонентною. На рисунку 1.8 показано процес формування сигналів в синфазному та квадратурному каналах.

Інформаційний потік $b_0(t)$ розбивається на пари біт, які відповідають одному символу модуляції та одній сигнальній точці сузір'я QPSK. Якщо парний біт (починаючи з нульового) дорівнює 1, то $I(t) > 1$ і навпаки.

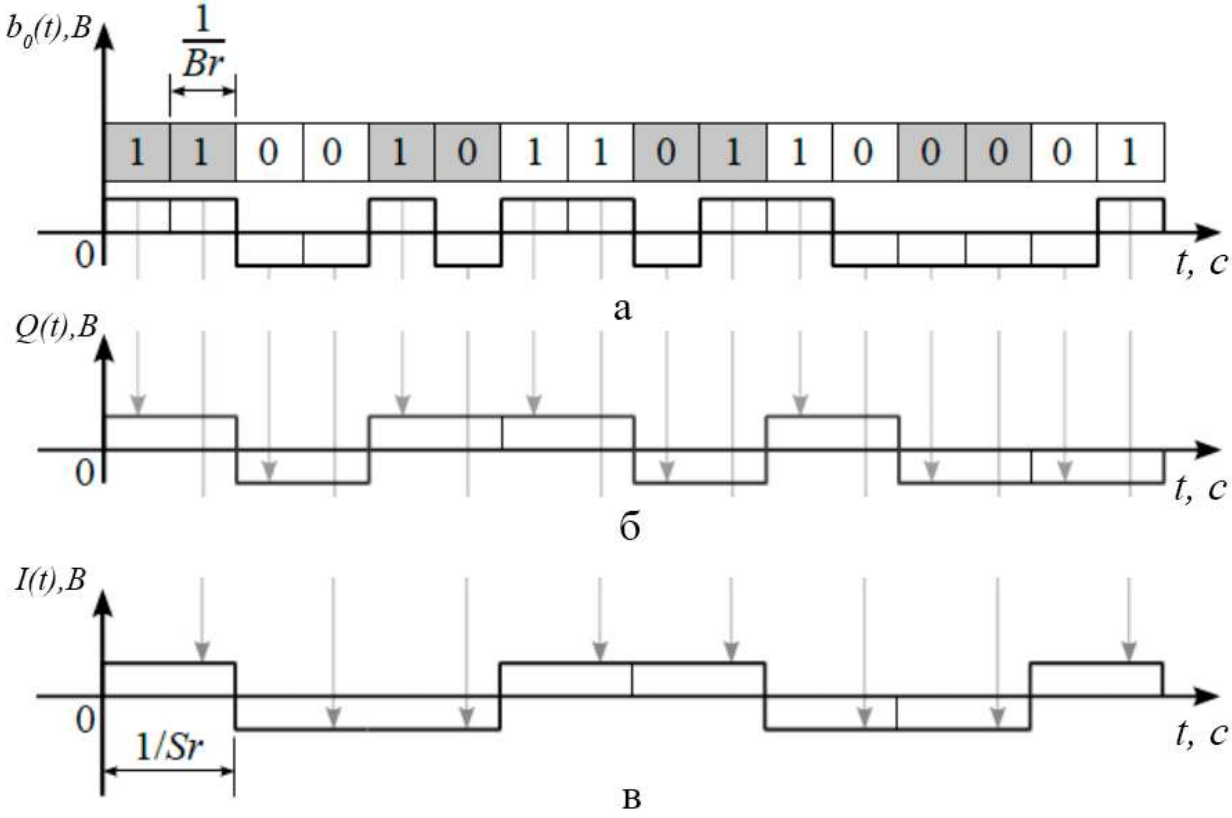

Рисунок 1.8 – Часові діаграми формування сигналів в синфазному та квадратурному каналах:
а – інформаційний потік; б – уявна складова сигналу; в – реальна складова сигналу

Аналогічно формується сигнал в квадратурному каналі. Процес формування синфазної та квадратурної компонент називається мапування. В залежності від пари біт потоку $b_0(t)$ на вході пристрою мапування, на виході маємо постійні на час тривалості цієї пари біт сигнали $I(t)$ та $Q(t)$, значення котрих залежать від інформації, що передається. Фазова огинаюча на виході QPSK модулятора показана на рисунку 1.9. На рисунку потік $b_0(t)$ поступає на пристрій мапування, яке формує синфазні та квадратурні компоненти, котрі далі модулюються несучим коливанням, рознесеним на $\pi/2$ радіан по фазі.

QPSK сигнал на виході модулятора має вигляд:

$$S_{qpsk}(t) = I(t) \cdot \cos(\omega_0 \cdot t + \phi_0) + Q(t) \sin(\omega_0 \cdot t + \phi_0)\,. \qquad (1.21)$$

Синфазна та квадратурна складові сигналу представляють собою відповідно дійсну та уявну частини комплексної огинаючої і є вихідними сигналами квадратурного модулятора:

$$z(t) = I(t) + j \cdot Q(t). \quad (1.22)$$

Таким чином, можна представити QPSK сигнал через його комплексну огинаючу [6]:

$$S_{qpsk}(t) = Re[z(t) \cdot \exp(j \cdot \omega_0 \cdot t)]. \quad (1.23)$$

З комплексної огинаючої можна виразити наступну фазову огинаючу:

$$\phi(t) = arctg\left(\frac{Im[z(t)]}{Re[z(t)]}\right) = arctg\left(\frac{Q(t)}{I(t)}\right). \quad (1.24)$$

Також на рисунку 1.9 зображені часові діаграми, які показують формування фазової огинаючої за синфазною та квадратурною компонентами.

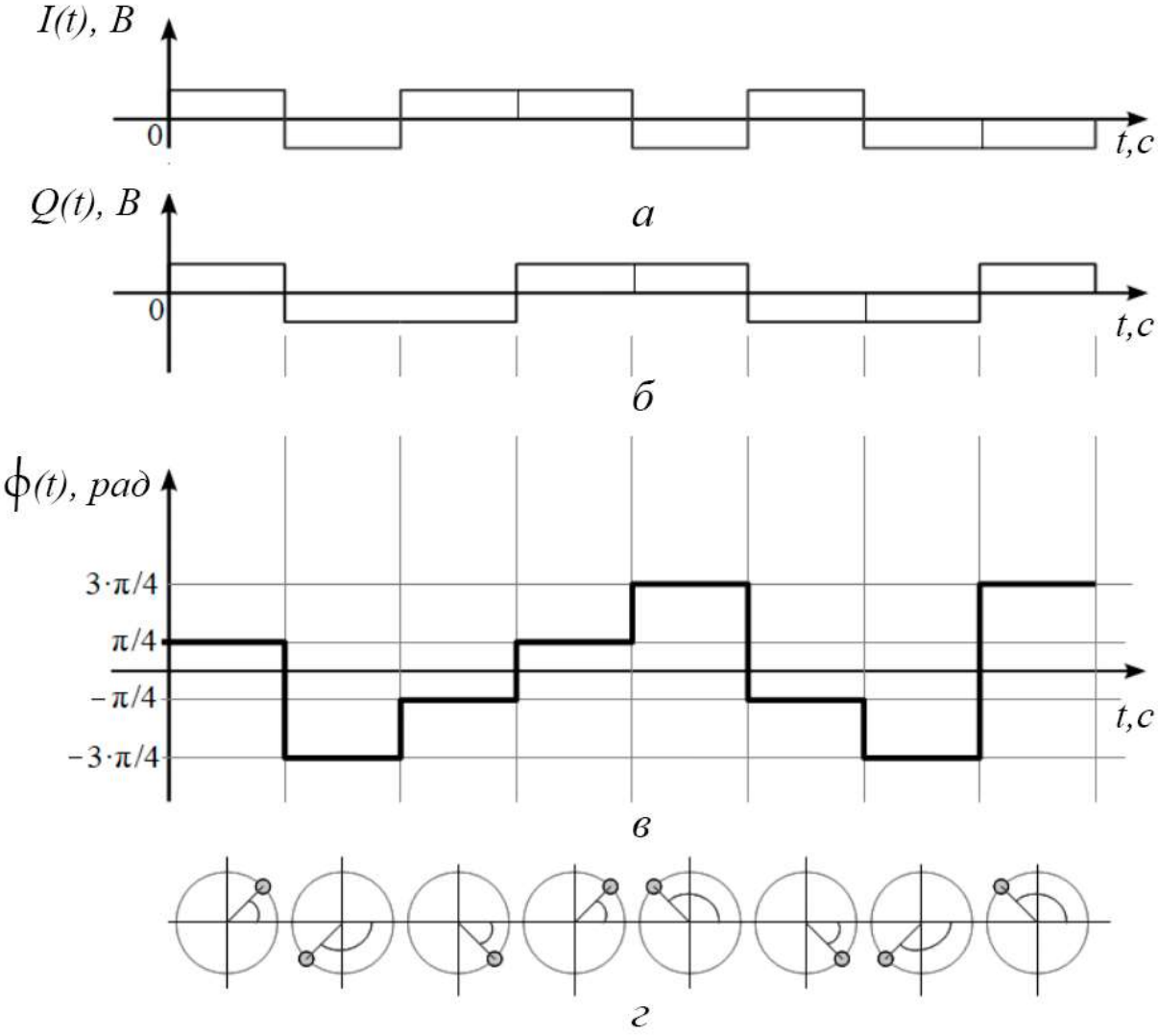

Рисунок 1.9 – часові діаграми фазової огинаючої сигналу:
а – синфазна компонента сигналу;
б – квадратурна компонента сигналу;
в – фазова огинаюча сигналу;
г – значення фази на комплексній площині

Фазова огинаюча QPSK сигналу представляє собою ступінчату функцію часу, яка має розриви в моменти зміни символів модуляції.

У 8-PSK модуляції, також використовуваній в системах супутникового зв'язку DVB-S2 та DVB-S2X, у відповідність кожному можливому набору з трьох біт інформаційного потоку ставиться один символ модуляції, відповідний одному з восьми можливих значень фази несучого коливання. Сузір'я сигнальних точок 8-PSK модуляції показано на рисунку 1.10.

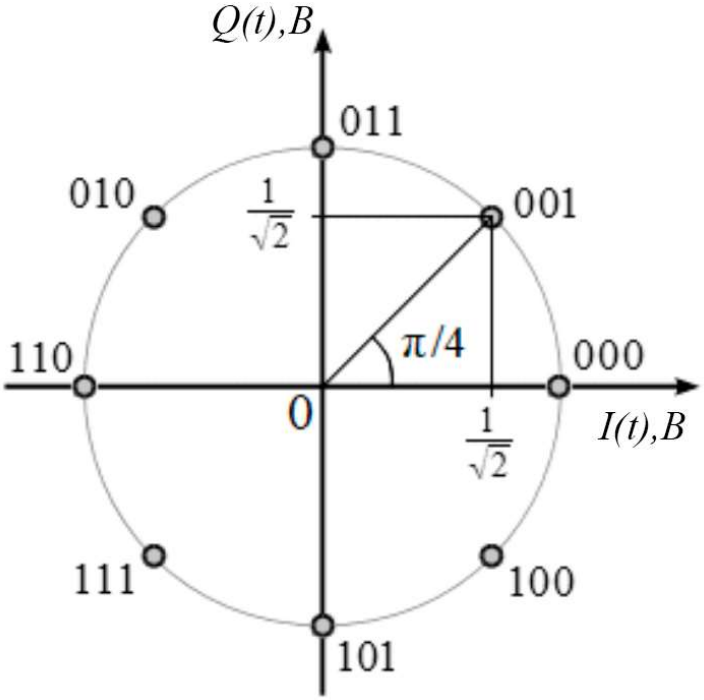

Рисунок 1.10 – Сузір'я сигнальних точок 8-PSK модуляції

Кожному з трьох слідуючих біт під час мапування ставляться у відповідність значення синфазної та квадратурної складових, які надалі модулюють коливання, рознесені по фазі на $\pi/2$ радіан.

1.3.2 M-APSK модуляція

Подальше збільшення позиційності при використанні модуляції M-PSK для збільшення спектральної ефективності призводить до значного зменшення завадостійкості. Це пов'язано з тим, що рознос по фазі, а, отже, і відстань помилки, зменшується при збільшенні позиційності M. З ціллю збереження високої завадостійкості при збільшенні спектральної ефективності в радіолініях зв'язку з космічними апаратами використовуються методи амплітудно-фазової модуляції M-APSK (англ. – Amplitude and Phase-Shift Keying – APSK). В системах DVB-S2 позиційність такої модуляції може досягати 32, а в DVB-S2X – 256.

В модуляції M-APSK інформацію несе не тільки фаза сигналу, а і його амплітуда. Наприклад, при модуляції 16-APSK використовуються дві окружності з сигнальними точками, на внутрішній з який знаходиться 4 сигнальні точки, а на зовнішній – 16 сигнальних точок[7], [8]. За рахунок цього досягається рівномірний розподіл сигнальних точок на амплітудно-фазовій

площині, тобто зменшується середня відстань помилки між різними сусідніми сигнальними точками. Як і в інших видах квадратурної модуляції, сигнал M-APSK може бути описаний як

$$S_{M-APSK}(t) = I(t) \cdot \cos(\omega_0 \cdot t + \phi_0) + Q(t) \cdot \sin(\omega_0 \cdot t + \phi_0). \quad (1.25)$$

1.3.3 Сигнальне сузір'я M-APSK модуляції

Великий науковий інтерес в області досліджень багатомірних сигнальних сузір'їв виник у вісімдесяті роки XX віку. На сьогоднішній день в більшості сучасних систем зв'язку використовуються багатопозиційні схеми модуляції з двомірними сигнальними сузір'ями.

Сигнальне сузір'я M-APSK модуляції складається з кількох концентричних окружностей. Число цих окружностей залежить від позиційності модуляції і обирається із розрахунку максимізації середньої відстані помилки при фіксованій максимальній амплітуді сигналу. Сигнал M-APSK може бути представлено у вигляді:

$$U_c(t) = A_i(t) \cos\left(\omega_0 \cdot t + \varphi_j(t)\right), \quad (1.26)$$

де $A_i(t)$ – дискретне значення амплітуди, В;

$\varphi_j(t)$ – дискретне значення фазового кута, рад.

Кут може приймати одне з фіксованих значень φ_1, φ_2, …, φ_M, котрі знаходяться в межах від нуля до 2π.

$$\varphi_j(t) = \frac{2\pi k}{M_i}, k = 1,2, \dots, M_i, \quad (1.27)$$

де k – рівноімовірні значення фаз;

M_i – кількість точок на окружності радіусом $A_i(t)$.

На рисунку 1.11 та рисунку 1.12 показані сигнальні сузір'я M-APSK, де $M = 16$ та $M = 32$ відповідно. Такі види позиційності використовуються в системах DVB-S2. В розширенні стандарту DVB-S2X використовуються й інші сузір'я аж до 256-APSK. Також для різних сценаріїв передбачені сузір'я сигнальних точок 16-APSK та 32-APSK з іншими конфігураціями точок й окружностей.

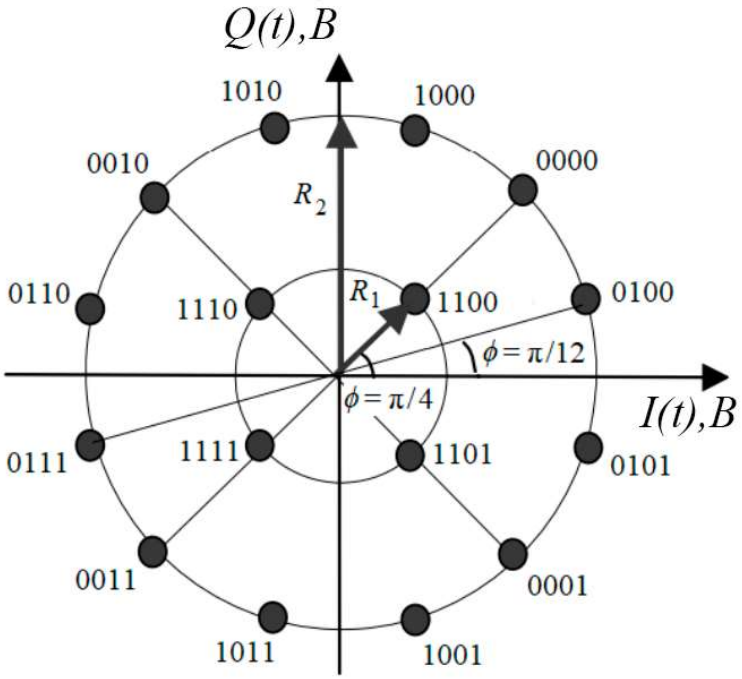

Рисунок 1.11 – Сигнальне сузір’я точок 16-APSK модуляції

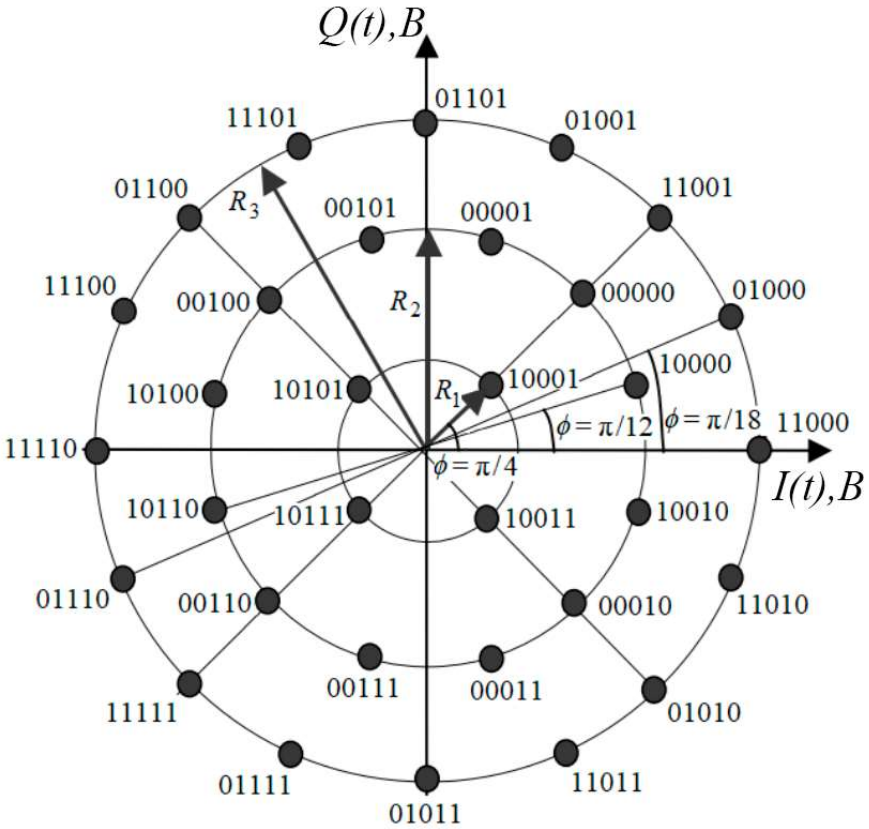

Рисунок 1.12 – Сигнальне сузір’я точок 32-APSK модуляції

1.4 Перспективні методи завадостійкого кодування інформації рекомендовані для радіоліній зв'язку з космічними апаратами

1.4.1 LDPC – кодування даних

Коди з низькою щільністю перевірок на – це клас лінійних блокових кодів з декодерами, що можуть бути імплементовані у пристрій, та які забезпечують високу продуктивність на великому наборі каналів передачі та зберігання даних. LDPC коди були винайдені Галлагером у своїй докторській дисертації 1960 р. [9] і в основному були проігноровані протягом 35 років. Визначним винятком є важлива робота Таннера у 1981 р. [10], в якій Таннер узагальнив LDPC коди та представив графічне представлення LDPC кодів, яке тепер називається графом Таннера. Вивчення LDPC кодів було відроджено в середині дев'яностих років завдяки роботі Маккея, Лубі та інших [11],[12], які помітили, незалежно від роботи Галлагера, переваги лінійних блокових кодів з розрідженими матрицями (матрицями з низькою щільністю) перевірок на парність.

1.4.1.1 Представлення LDPC кодів

LDPC коди можуть бути представлені двома способами: матричне представлення та графічне представлення. Розглянемо матричне представлення. Для простоти будуть розглянуті лише двійкові LDPC коди.

Коди з низькою щільністю перевірок на парність це лінійні блокові коди які задані перевірочною матрицею H розмірністю $m \times n$ з малою кількістю одиниць. Регулярний LDPC код – це лінійний блоковий код, перевірочна матриця H котрого має вагу стовбця g та вагу рядка r, де $r = g(n/m)$ та $g \ll m$. Якщо H має низьку щільність, але ваги стовбця та рядка не є постійними значеннями, такий код називається нерегулярним. З причин, що будуть описані далі, майже всі LDPC коди накладають додаткове структурну властивість на перевірочну матрицю: ніякі два рядка (або стовбця) не мають більше, ніж одну загальну позицію, яка містить ненульовий елемент. Ця властивість називається обмеженням рядок-стовбець.

Опис «низька щільність» дуже розпливчастий та не визначається кількісно, але щільність ненульових елементів, яка дорівнює або менше ніж 0.01 (1% або менше) може називатися низькою щільністю. Щільність ненульових елементів повинна бути достатньо низькою для того, щоб забезпечити ефективне ітеративне декодування. Це ключове нововведення, яке лежить в основі винайдення LDPC кодів.

Як добре відомо оптимальне (за максимумом правдоподібності) декодування загального лінійного блокового коду неможливе через його величезну складність. Але кодування з низькою щільністю перевірок на парність облегшує процес ітеративного декодування, яке зазвичай має ефективність близьку до максимальної правдоподібності, при частоті помилок, яка представляє інтерес для багатьох.

Як буде показано далі побудова LDPC коду зазвичай включає побудову перевірочної матриці H. Швидкість регулярного LDPC коду обмежена розмірами перевірочної матриці.

$$\left.\begin{aligned} R \geq 1 - \frac{m}{n}, \\ R \geq 1 - \frac{g}{r}. \end{aligned}\right\} \tag{1.28}$$

Графічне представлення LDPC коду – це граф Таннера. Граф Таннера для LDPC коду це як гратчаста діаграма для згорткового коду, в тому сенсі, що він повністю забезпечує повне представлення коду та допомагає в описі алгоритмів декодування. Граф Таннера – це двочастковий граф, вузли котрого можна розділити на два типи, з ребрами, які з'єднують вузли лише різних типів. Два типи вузлів в графі Таннера – це змінний вузол (вузол кодового біта) та перевірочний вузол.

Граф Таннера коду будується наступним чином: перевірочний вузол i з'єднується із змінним вузлом j всякий раз, коли елемент h_{ij} в перевірочній матриці H дорівнює одиниці. З цього правила помітимо, що в графі існує m перевірочних вузлів, по одному на перевірочне вираження, та n змінних вузлів, по одному на кожний кодовий біт.

Отже, m рядків перевірочної матриці визначають m з'єднань перевірочних вузлів, та n стовбців перевірочної матриці визначають n з'єднань змінних вузлів. Таким чином, допустимі n-бітні кодові слова представлені n змінних вузлів – це в точності кодові слова.

Розглянемо лінійний блоковий код (10,5) з вагою стовбця $w_c = 2$ та вагою рядка $w_c = 4$, і перевірочною матрицею, яку задано вираженням (1.29).

$$H = \begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix}. \tag{1.29}$$

Граф Таннера, який відповідає перевірочній матриці (1.29) показано на рисунку 1.13. Помітимо, що змінні вузли 0, 1, 2 та 3 з'єднані з перевірочним вузлом 0, у відповідності з тим, що в нульовому рядку перевірочної матриці $h_{00} = h_{01} = h_{02} = h_{03} = 1$ (всі решта дорівнюють нулю). Аналогічна ситуація спостерігається і для перевірочних вузлів під номерами 1, 2, 3 та 4, котрі відповідають рядкам перевірочної матриці з першого по четвертий.

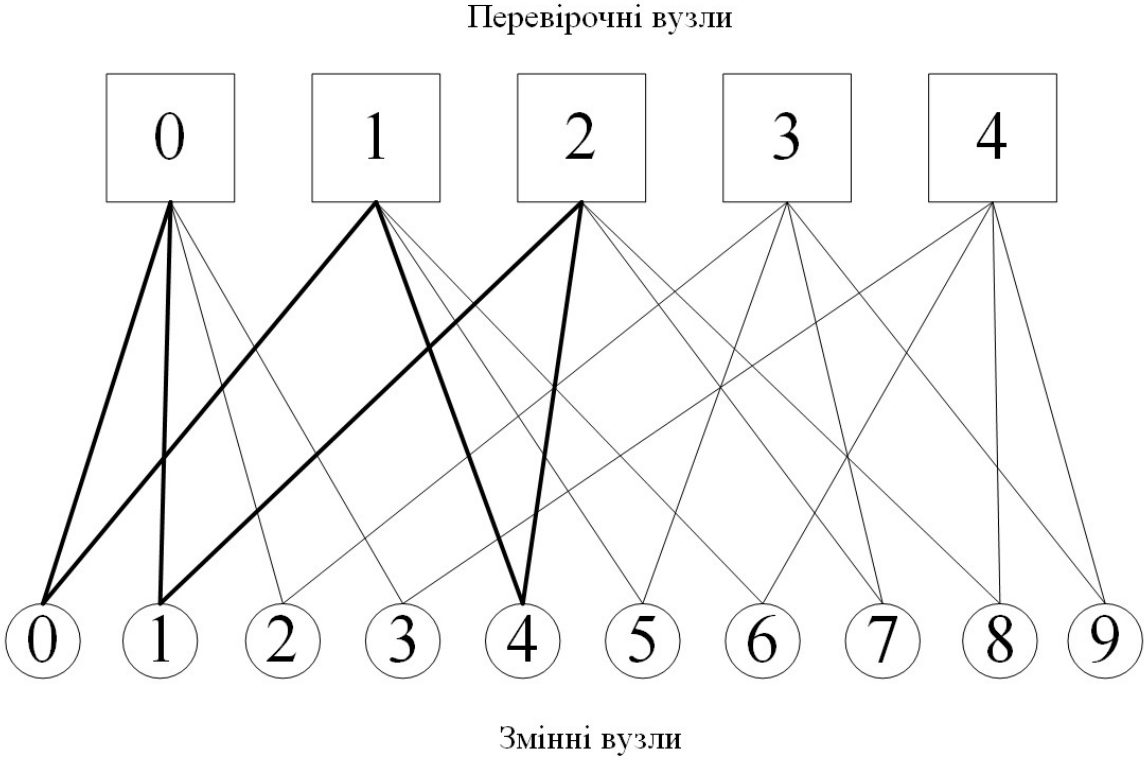

Рисунок 1.13 – Граф Таннера

Беручи до уваги те, що

$$vH^T = 0, \tag{1.30}$$

змінні вузли, які з'єднані одним і тим самим перевірочним вузлом в сумі по модулю 2 повинні давати результат, який дорівнює 0.

Граф Таннера в LDPC кодуванні діє як план для ітеративного декодеру наступним чином. Кожен з вузлів діє як локально працюючий процесор, а кожне з'єднання діє як шина, яка передає інформацію від одного вузла до його сусідів. Інформація, яка передається, зазвичай, представляє собою імовірнісну інформацію, наприклад, логарифмічні відношення правдоподібності (англ. – log-likelihood ratios – LLRs), які відносяться до значень бітів привласнених змінним вузлам. LDPC декодер ініціюється логарифмічними відношеннями правдоподібності з каналу передачі, які приймаються n змінних вузлів. На початку кожної пів-ітерації в базовому алгоритмі ітеративного декодування кожний змінний вузол приймає дані із каналу та кожного з його сусідів, та на основі цієї інформації розраховує вихідні дані для кожного із своїх сусідніх перевірочних вузлів. В наступній пів-ітерації кожен перевірочний вузол приймає вхідні дані від кожного змінного вузла, які з ним з'єднані, та розраховує вихідні дані для кожного з'єднаного з ним змінного вузла. Ітерації обміну даними між змінними вузлами та перевірочними вузлами повторюються до тих пір, поки не буде знайдено кодове слово або не буде досягнуто максимально доступна кількість ітерацій.

Ефективність ітеративного декодеру залежить від ряду структурних властивостей графу Таннера, на котрому побудовано декодер. На рисунку 1.13 послідовність із 6 наведених з'єднань утворюють замкнений шлях, який називається циклом. Короткі цикли погіршують продуктивність алгоритмів

ітеративного декодування, які використовуються LDPC декодерами. Цикли змушують декодер працювати локально в деяких частинах графу (наприклад, постійно протягом короткого циклу), тому глобальне оптимальне рішення неможливе. Також слід відмити необхідність матриці саме з низькою щільністю перевірок на парність, так, при високій щільності (половина матриці заповнена одиницями) буде існувати велика кількість коротких циклів, що виключає використання ітеративного декодеру.

Довжина циклу дорівнює кількості ребер, які утворюють цикл, тому довжина циклу на рисунку 1.13 дорівнює 6. Цикл довжиною l часто називають l-циклом. Мінімальна довжина циклу в даному двобічному графі називається обхватом. Обхват графу Таннера у прикладі складає 6. Найкоротший можливий цикл у двобічному графі явно являє собою цикл довжиною 4, і такі цикли проявляються в перевірочній матриці як чотири одиниці, що лежать на чотирьох кутах прямокутної підматриці.

Граф Таннера, наведений у прикладі – регулярний: кожен змінний вузол має два крайових з'єднання, і кожен перевірочний вузол має чотири крайових з'єднання. Говорять, що ступінь кожного змінного вузла в такому випадку дорівнює 2, а ступінь кожного перевірочного вузла – 4. З цього прикладу також видно, що $mr = ng$ має виконуватися для всіх регулярних LDPC кодів, оскільки mr і ng дорівнюють кількості ребер на графі.

За допомогою нерегулярних LDPC кодів можна ближче наблизитись до пропускної здатності, ніж із регулярними LDPC кодами. Для нерегулярних LDPC кодів параметри g та r замінюються залежно від стовбців та рядків, тому в такому випадку такі позначення не корисні. Натомість у літературі зазвичай вказують поліноми розподілу ступеня змінних та перевірочних вузлів, що позначаються відповідно $\lambda(X)$ та $\rho(X)$.

$$\lambda(X) = \sum_{d=1}^{d_v} \lambda_d X^{d-1}, \tag{1.31}$$

де λ_d – позначає частку всіх ребер, приєднаних до змінного вузла ступеня d_v;

d_v – максимальний ступінь змінного вузла.

$$\rho(X) = \sum_{d=1}^{d_c} \rho_d X^{d-1}, \tag{1.32}$$

де ρ_d – позначає частку всіх ребер, приєднаних до перевірочного вузла ступеня d_v;

d_v – максимальний ступінь перевірочного вузла.

Зауважимо, що для регулярного коду вище, для якого $g = 2$ та $r = 4$, маємо $\lambda(X) = X$ та $\rho(X) = X^3$.

Позначимо кількість змінних вузлів ступеня d через $N_v(d)$, а кількість перевірочних вузлів ступеня d через $N_c(d)$. Також позначимо через E кількість ребер на графі. Тоді можна записати

$$E = \frac{n}{\int_0^1 \lambda(X)dX} = \frac{m}{\int_0^1 \rho(X)dX}, \tag{1.33}$$

$$N_v(d) = \frac{E\lambda_d}{d} = \frac{\frac{n\lambda_d}{d}}{\int_0^1 \lambda(X)dX}, \tag{1.34}$$

$$N_c(d) = \frac{E\rho_d}{d} = \frac{\frac{m\rho_d}{d}}{\int_0^1 \rho(X)dX}. \tag{1.35}$$

З двох виразів для E можна зробити висновок, що швидкість коду обмежена як

$$\left.\begin{aligned} R &\geq 1 - \frac{m}{n}, \\ R &\geq 1 - \frac{\int_0^1 \rho(X)dX}{\int_0^1 \lambda(X)dX}. \end{aligned}\right\} \tag{1.36}$$

Поліноми $\lambda(X)$ та $\rho(X)$ представляють розподіл ступенів графу Таннера з «крайової перспективи». Розподіл ступенів також може бути представлений з точки зору вузла, використовуючи позначення $\tilde{\lambda}(X)$ та $\tilde{\rho}(X)$, де коефіцієнт $\tilde{\lambda}(X)$– частка всіх змінних вузлів, що мають ступінь d, та $\tilde{\rho}(X)$ – частка всіх перевірочних вузлів, що мають ступінь d. Це можна показати так:

$$\tilde{\lambda}(X) = \frac{\frac{\lambda_d}{d}}{\int_0^1 \lambda(X)dX}, \tag{1.37}$$

$$\tilde{\rho}(X) = \frac{\frac{\rho_d}{d}}{\int_0^1 \rho(X)dX}. \tag{1.38}$$

1.4.1.2 Класифікація LDPC кодів

Перейдемо до класифікації LDPC кодів. Оригінальні LDPC коди є випадковими у тому сенсі, що їх матриці перевірки на парність мають мало структури. Це проблематично тим, що і кодування, і декодування стають досить складними, коли код не має структури, крім того, що є лінійним кодом. Зовсім недавно були побудовані LDPC коди зі структурою.

Найбільш очевидний тип структури – це циклічний код. Кодер циклічного коду складається з єдиного регістру зсуву довжини $n - k$, декількох двійкових суматорів та затвору. Номінальною матрицею перевірки парності H циклічного коду є циркулянт $n \times n$, тобто кожен рядок – це циклічний зсув того рядка, що над ним, причому перший рядок – це циклічний зсув останнього рядка. Вплив розрідженої циркулянтної матриці H на складність LDPC декодера є значним: оскільки кожне рівняння перевірки на парність тісно пов’язане зі своїм попередником та його наступником, тому реалізація декодеру може бути значно

спрощена, порівняно з випадковими розрідженими H матрицями, в яких довільно прокладені з'єднання між змінними та перевірочними вузлами.

Однак, крім регулярності, недоліком циклічних LDPC кодів є те, що номінальна матриця H має розмірність $n \times n$, незалежно від швидкості кодування, що означає більш складний декодер. Іншим недоліком є те, що відомі циклічні LDPC коди, як правило, мають великі ваги рядків, що робить реалізацію декодера ще складнішою.

З іншого боку, циклічні LDPC коди мають, як правило, великі мінімальні відстані та дуже низькі ітеративно декодовані мінімальні рівні помилок.

Квазіциклічні коди також мають структуру, що призводить до спрощених конструкцій кодерів та декодерів. Крім того, вони забезпечують більшу гнучкість у розробці коду, особливо нерегулярність, і, отже, призводять до вдосконалення кодів порівняно з циклічними кодами. Перевірочна матриця квазіциклічного коду зазвичай представляється у вигляді масиву циркулянтів, наприклад, як показано у виразі (1.39).

$$H = \begin{bmatrix} A_{11} \dots A_{1N} \\ A_{M1} \dots A_{MN} \end{bmatrix}, \tag{1.39}$$

де A_{MN} – матриця циркулянт розмірністю $Q \times Q$.

Для LDPC кодів циркулянти повинні бути розрідженими, та позначення циркулянт ваги 1, означає, що вага кожного рядка та стовбця циркулянту є однаковою.

Для досягнення нерегулярності деякі з циркулянтів мають бути матрицею $Q \times Q$ повністю заповненою нулями, використовуючи техніку, яка називається маскуванням.

На додаток до розподілу LDPC кодів на класи – циклічний, квазіциклічний та випадковий (але лінійний), методи побудови LDPC коду також можуть бути розділені. Перший клас технік побудови можна охарактеризувати як алгоритмічний або комп'ютерний. Другий клас технік побудови складається з тих, що базуються на кінцевій математиці, включаючи алгебру, комбінаторику та теорію графів. Зазначають, що комп'ютерні методи побудови можуть призвести до випадкових або структурованих кодів. Математичні методи побудови зазвичай ведуть до сруктурованих LDPC кодів, хоча існують винятки.

Під час розгляду ітеративного декодування корисно розглянути узагальнення LDPC кодів. Таннер був першим, хто узагальнив ідею кодів з низькою щільністю перевірок на парність Галагера у своїй новаторській роботі [10], в якій були представлені графи Таннера. На графі Таннера узагальненого LDPC коду показаного на рисунку 1.14, перевірочні вузли є більш загальними, ніж обмеження для єдиної перевірки парності. Насправді, змінні вузли можуть представляти більш складні коди, ніж коди повторення, які вони представляють у стандартних LDPC кодах [13].

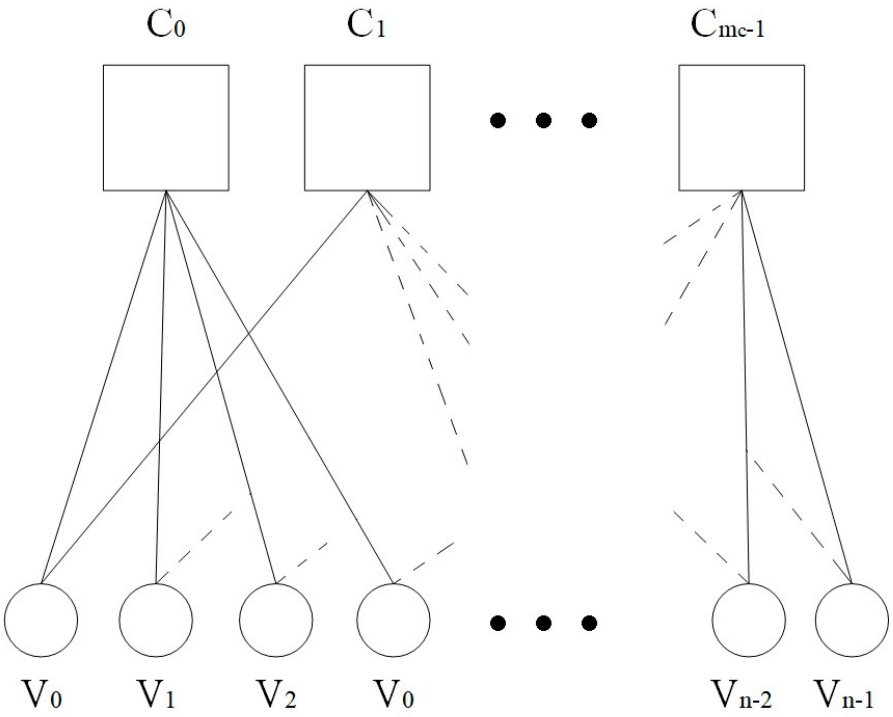

Рисунок 1.14 – Граф Таннера узагальненого LDPC коду

1.4.1.3 Передача повідомлень та турбо принцип

Ключовим нововведенням LDPC кодів є характер перевірочної матриці з низькою щільністю перевірок на парність, що полегшує ітеративне декодування. Хоча різні ітеративні алгоритми декодування є неоптимальними, вони часто забезпечують майже оптимальну продуктивність, принаймні для у процентному підрахунку помилок. Зокрема, так званий алгоритм суми добутків (англ. – Sum-Product Algorithm – SPA) є загальним алгоритмом, який забезпечує майже оптимальну продуктивність у широкому класі каналів.

Для досягнення мети представлення майже оптимального SPA для ітеративного декодування LDPC кодів корисно спочатку розглянути картину більш загального декодування за допомогою передачі повідомлень. Декодування за допомогою передачі повідомлень відноситься до набору декодерів низької складності, що працюють розподіленим чином, щоб декодувати прийняте кодове слово в об’єднаній схемі кодування. Варіанти SPA та інші алгоритми декодування за допомогою передачі повідомлень були винайдені незалежно в ряді різних контекстів, включаючи алгоритм розповсюдження довіри (англ. – в Belief-Propagation – BP) для використання в Байєсівських мережах та турбо-декодування для паралельно з’єднаних згорткових кодів (які є оригінальними турбо-кодами).

Дослідники уподібнюють алгоритм розповсюдження довіри – розподіленому підрахунку солдатів, а винахідники турбо-кодів були натхненні двигунами з турбонаддувом, в яких двигун отримує збільшену кількість повітря, використовуючи власні вихлопи у зворотному зв’язку.

LDPC коди можна розглядати як узагальнену конкатенацію багатьох кодів з однією перевіркою на парністю. Декодер з передачею повідомлень для LDPC

коду використовує індивідуальний декодер для кожного коду з однією перевіркою на парність, і ці декодери працюють спільно розподіленим чином, щоб визначити правильні бітові значення коду. Як зазначено вище, графи Таннера часто служать моделями декодерів. У контексті декодера, що передає повідомлення, перевірочні вузли представляють декодери з однією перевіркою на парність, які працюють спільно для декодування отриманого слова.

Розглянемо приклад роботи такого декодеру. На рисунку 1.15 показано масив коду, який є продуктом конкатенації кодів з однією перевіркою на парність, та в результаті дає LDPC код. В 1.40 показано перевірочну матрицю такого коду, та на рисунку 1.16 показано граф Таннера, який відповідає перевірочній матриці коду. Кожен перевірочний вузол відповідає рядку в перевірочній матриці. Верхні перевірочні вузли представляють декодери по рядкам для кодів з однією перевіркою на парність, що входять до масиву на рисунку 1.15, а нижні перевірочні вузли представляють декодери по стовбцям.

c_1	c_2	c_3
c_4	c_5	c_6
c_7	c_8	c_9

Рисунок 1.15 – Масив кодів з однією перевіркою на парність як LDPC код

$$H = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 \end{bmatrix}. \quad (1.40)$$

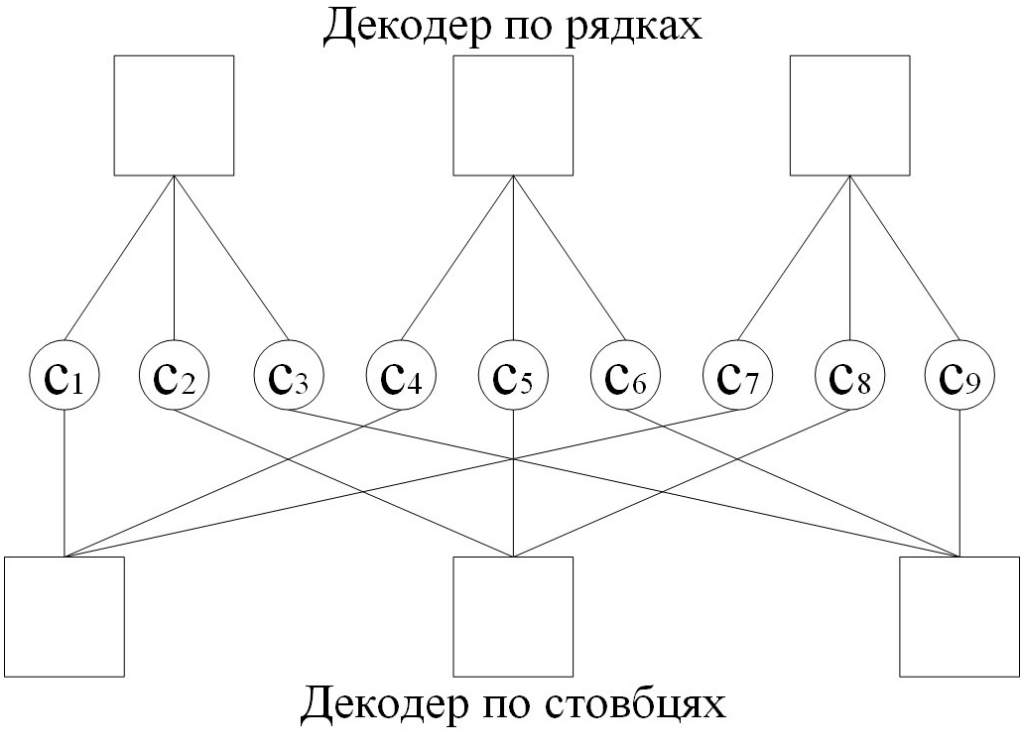

Рисунок 1.16 – Графічна модель декодера

На рисунку 1.17 представлений приклад декодування у двійковому каналі зі стираннями, який можна прийняти за стандартне ітераційне декодування по рядках / по стовбцях у двійковому каналі зі стираннями, техніку, відому вже декілька десятиліть.

e	e	1	e	e	1	1	e	1	1	0	1
1	e	0	1	1	0	1	1	0	1	1	0
0	e	e	0	e	e	0	e	1	0	1	1

а б в г

Рисунок 1.17 – Приклад декодування:
а – прийняте кодове слово;
б – після декодування по рядках;
в – після декодування по стовбцях;
г – після декодування по рядках

На прикладі, оскільки кожен рядок і стовбець масиву коду відповідає кодовому слову з однією перевіркою на парність, та середнє стирання в отриманому слові має дорівнювати 1, оскільки $1 + e + 0 = 0(mod\ 2)$. Слід звернути увагу на те, як на рисунку 1.17 видно, що певні стирання можуть бути вирішені лише декодером по рядках, а інші стирання можуть бути вирішені лише декодером по стовбцях, але спільною роботою всі стирання врешті-решт вирішуються. Також можна еквівалентно розглянути це декодування з точки

зору графу Таннера, в якому компоненти однієї перевірки на парність ітераційно вирішують стирання: спочатку верхні декодери, а потім нижні декодери.

Ця точка зору передачі повідомлень розширюється за допомогою аналогії кросворду. При вирішенні кросворду, коли людина перебирає слова «через» і «вниз», відповіді в одному напрямку все частіше допомагають знайти відповіді в іншому напрямку. Звичайно, важче спершу знайти всі слова «через», а потім усі слова «вниз», ніж робити ітерацію вперед і назад, отримуючи підказки (повідомлення) від пересічних слів.

Розглянемо остаточний ілюстративний приклад ідеї передачі повідомлень у системі розподіленого процесора. Цей приклад дозволяє ввести важливе поняття зовнішньої інформації. Це також чітко пояснює (без доказів) той факт, що декодування за допомогою передачі повідомлень для колекції складових декодерів, розташованих у графі, є оптимальним за умови, що граф не містить циклів, але не є оптимальним для графів з циклами. Прикладом може слугувати відома проблема підрахунку солдат у літературі Байєса[14].

Зобразимо шість солдатів у лінійному формуванні (рисунок 1.18 а). Мета полягає в тому, щоб кожен із солдатів дізнався загальну кількість присутніх солдатів шляхом розподіленого підрахунку. Правила підрахунку наступні: кожен солдат отримує номер з одного боку, додає один для себе і передає суму на інший бік. Солдати на кінцях отримують нуль збоку без сусіда. Сума числа, яке солдат отримує з одного боку, і числа, яке солдат передає на інший бік, дорівнює загальній кількості солдатів. Звичайно, ця сума має значення лише з одного боку для солдатів на кінцях.

Зобразимо формування у вигляді дерева (рисунок 1.18 б). Слід звернути увагу, що третій солдат зліва отримує повідомлення від трьох сусідніх солдатів, і тому правила повинні бути змінені для цієї більше загальної ситуації. Модифікований набір правил підрахунку такий: повідомлення про те, що довільний солдат X переходить до довільного сусіднього солдату Y, дорівнює сумі всіх вхідних повідомлень, плюс одне для солдата X, мінус одне повідомлення, яке солдат Y щойно надіслав солдату X. Солдати на кінцях отримують нуль збоку без сусіда. Сума числа, яке солдат отримує від будь-якого зі своїх сусідів, плюс число, яке солдат передає цьому сусідові, дорівнює загальній кількості солдатів (рисунок 1.18 б).

Це правило передачі повідомлень вводить поняття зовнішньої інформації. Ідея полягає в тому, що солдат не передає сусідньому солдату жодної інформації, яку вже має сусідній солдат, тобто передається лише зовнішня інформація. Саме з цієї причини солдат Y отримує загальну кількість повідомлень, яка має солдат X, за вирахуванням інформації, яку вже має солдат Y. Говорять, що солдат X передає солдату Y лише зовнішню інформацію, яка може бути обчислена як

$$I_{X\to Y} = \sum_{Z\in N(X)} I_{Z\to X} - I_{Y\to X} + I_X = \sum_{Z\in N(X)-\{Y\}} I_{Z\to X} + I_X\,, \qquad (1.41)$$

де $N(X)$ – сукупність сусідів солдата X;

$I_{X\to Y}$ – зовнішня інформація, що надсилається від солдата X солдату Y;

$I_{Z\to X}$ – зовнішня інформація, що надсилається від солдата Z солдату X;

$I_{Y\to X}$ – зовнішня інформація, що надсилається від солдата Y солдату X;

I_X – внутрішня інформація.

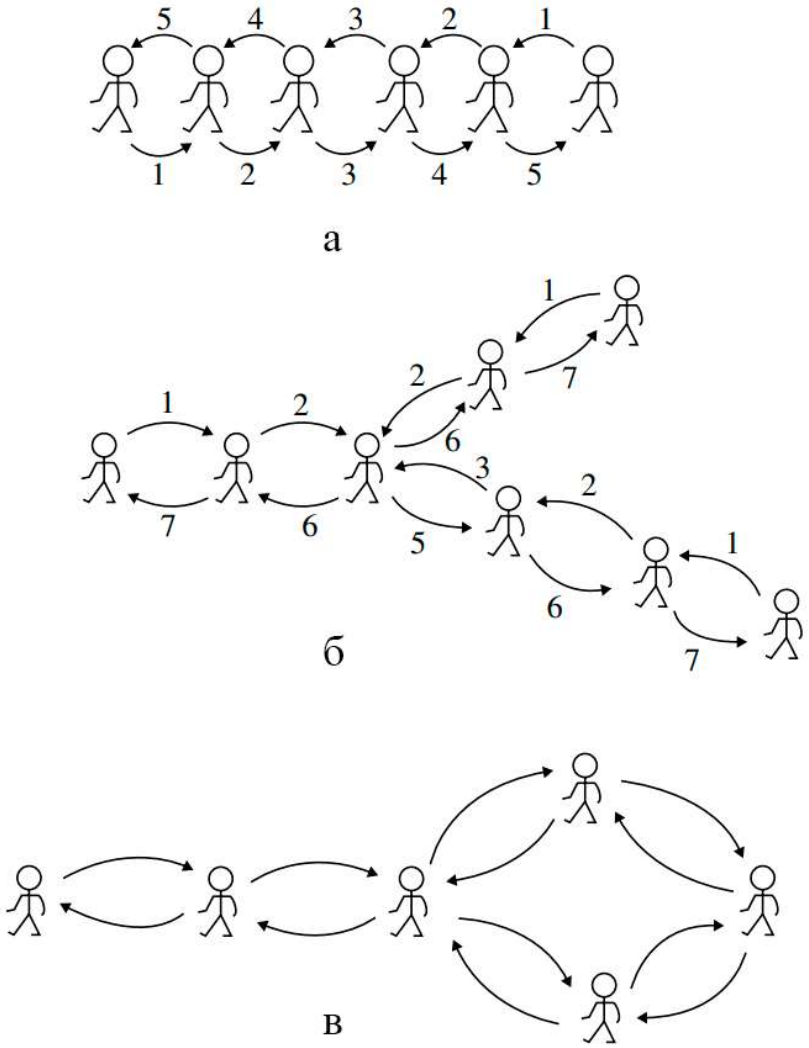

Рисунок 1.18 – Розподілений підрахунок солдатів:
а – лінійне формування;
б – деревинне формування;
в – формування з циклом

Нарешті, розглянемо формування на рисунку 1.18 (в), яке містить цикл. Можна легко перевірити, що така ситуація неможлива: незалежно від того, яке правило підрахунку можна розробити, цикл представляє тип позитивного зворотного зв’язку як за годинниковою стрілкою, так і проти годинникової стрілки, так що повідомлення, передані в циклі, будуть збільшуватись без

обмежень, поки проходження по циклу будуть тривати. Цей приклад демонструє, що повідомлення що передається по графу, не може бути визнано оптимальним, якщо граф містить один або кілька циклів. Однак, хоча більшість практичних кодів містять цикли, добре відомо, що декодування за допомогою передачі повідомлень виконується дуже добре для правильно розроблених кодів для більшості рівнів помилок.

Описане вище поняття передачі зовнішньої інформації було назване турбо принципом в контексті ітеративного декодування об'єднаних кодів у каналах зв'язку. Зображення турбо принципу наведено на рисунку 1.19. На рисунку,

$$I_X^{\text{рез}} = \sum_{Z \in N(X)} I_{Z \to X} + I_X \,. \tag{1.42}$$

І аналогічно для $I_Y^{\text{рез}}$. I_X та I_Y представляють внутрішню інформацію, отриману з каналу. Слід звернути увагу, що рисунку 1.19 – це лише блок-схема рівняння (1.41).

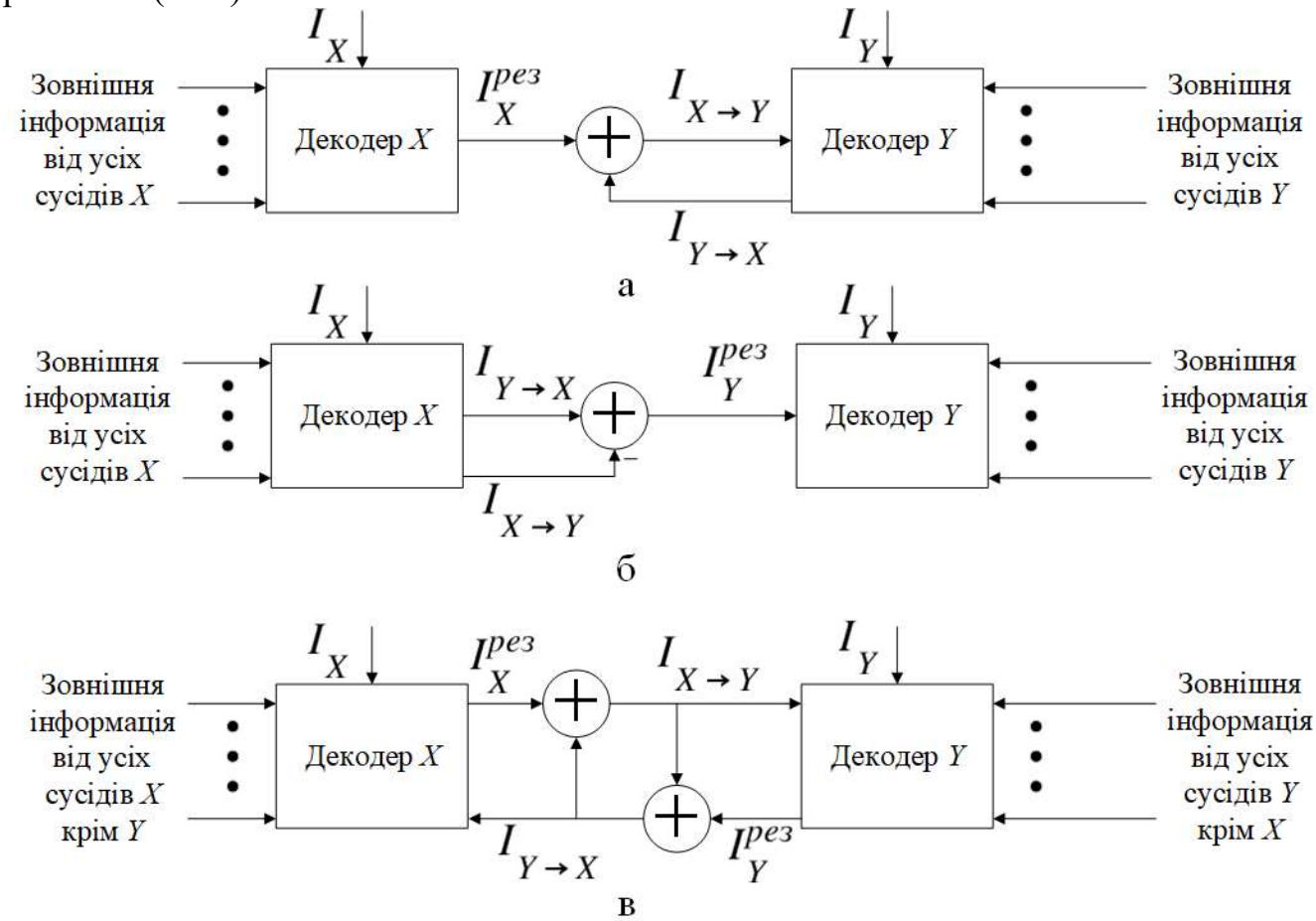

Рисунок 1.19 – Турбо принцип для об'єднаних схем кодування:
а – зовнішня інформація від декодера X до декодера Y;
б – зовнішня інформація від декодера Y до декодера X;
в – симетрична комбінація

Повідомлення буде передаватися від декодера X до декодера Y (рисунок 1.19 а) багато разів на практиці, хоча багато залежить від обраної схеми

планування, яка координує повідомлення, що передається між різними декодерами.

1.4.1.4 Алгоритм суми добутків

У цьому розділі буде розглянуто алгоритм суми добутків для загальних двійкових каналів без пам'яті, застосовуючи турбо принцип. Як особливі випадки будуть розглянуті двійкові канали зі стираннями, симетричні канали, двійкові канали з адитивним білим Гаусовим шумом та незалежний канал затухання Релея.

На додаток до впровадження LDPC кодів у своїй основній роботі в 1960 році, Галлагер також запровадив майже оптимальний алгоритм декодування, який тепер називається алгоритмом суми добутків. Цей алгоритм також іноді називають алгоритмом поширення довіри, ім'я взяте з літератури Байєса, де алгоритм було виведено незалежно [14]. Алгоритм ідентичний для всіх каналів без пам'яті, тому розгляд буде загальним.

Критерієм оптимальності, що лежить в основі розробки декодера, який використовує алгоритм, є символьний максимум апостеріорної імовірності. Подібно до оптимального декодування символів у згоркових кодах (тобто BCJR алгоритму [22], [23]). Алгоритм націлений на обчислення апостеріорної імовірності того, що конкретний біт у переданому кодовому слові $v = [v_0, v_1, \dots, v_{n-1}]$ дорівнює одиниці, враховуючи отримане слово $y = [y_0, y_1, \dots, y_{n-1}]$. Не втрачаючи загальності, зосередимося на декодуванні біта v_j, так що ми зацікавлені в обчисленні апостеріорної імовірності,

$$\Pr\left(v_j = 1 \middle| y\right). \tag{1.43}$$

Апостеріорна імовірність в такому випадку:

$$l\left(v_j \middle| y\right) = \frac{\Pr\left(v_j = 0 \middle| y\right)}{\Pr\left(v_j = 1 \middle| y\right)}, \tag{1.44}$$

або більш чисельно стабільний логарифмічна апостеріорна імовірність, також звана логарифмічним коефіцієнтом правдоподібності (КП),

$$L\left(v_j \middle| y\right) = \log\left(\frac{\Pr\left(v_j = 0 \middle| y\right)}{\Pr\left(v_j = 1 \middle| y\right)}\right). \tag{1.45}$$

Алгоритм суми добутків для обчислення $\Pr\left(v_j = 1 \middle| y\right)$, $l(v_j|y)$, або $L(v_j|y)$ є розподіленим алгоритмом, який застосовує турбо принцип до графу Танера коду [15]. Це легко побачити на рисунку 1.20, де представлено інший варіант графу Таннера на рисунку 1.13. Інший варіант графу Таннера робить більш очевидним уявлення про те, що LDPC код можна вважати набором кодів з однією перевіркою на парність, об'єднаних через перемежувач до набору кодів повторення. Крім того, коди з однією перевіркою на парність розглядають як зовнішні коди, тобто як ті, котрі не підключені до каналу. Ребра, що відходять від змінних вузлів на рисунку 1.20 підключені до каналу.

Можна застосувати турбо принцип на рисунку 1.19 до об’єднання кодів повторення та кодів з однією перевіркою на парність наступним чином.

На рисунку 1.21 зображена ситуація з декодером кодів повторення (змінні вузли) для змінних вузлів зі степенем, що перевищує 2 степінь змінних вузлів на рисунку 1.20.

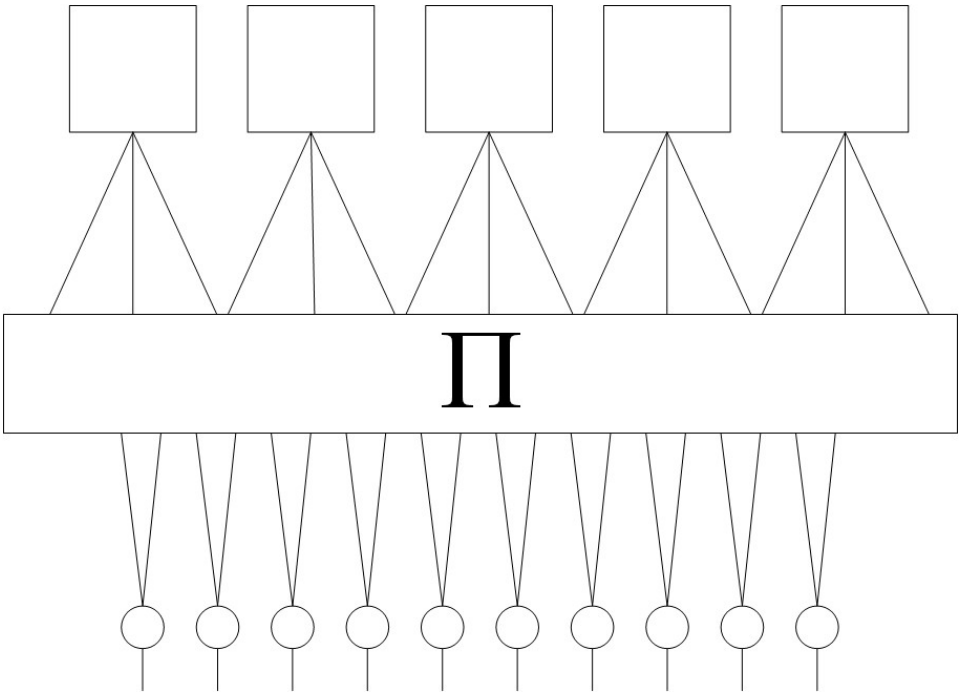

Рисунок 1.20 – Графічне представлення LDPC коду

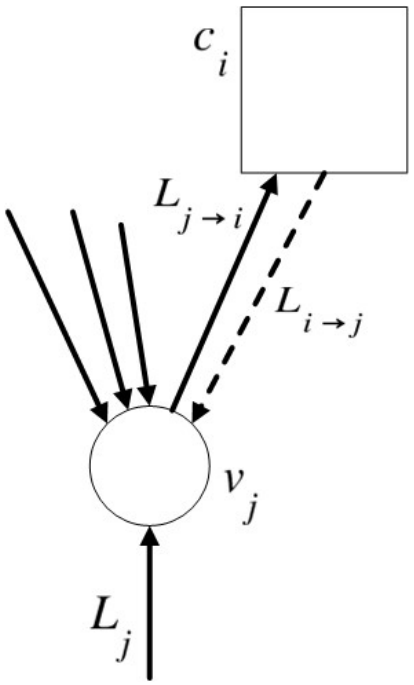

Рисунок 1.21 – Декодер змінних вузлів (декодер кодів повторення)

Слід звернути увагу, що j декодер змінних вузлів приймає КП як від каналу, так і від своїх сусідів. Однак при обчисленні зовнішньої інформації $L_{j\to i}$, j змінному вузлу не потрібно отримувати $L_{i\to j}$ від перевірочного i вузла, оскільки вона все одно віднімається за рівнянням (1.41) або згідно з рисунком 1.19. На

рисунку 1.22 показано ситуацію з декодером кодів з однією перевіркою на парність (перевірочний вузол). Подібно до випадку з змінним вузлом, при обчисленні $L_{i\to j}$ для деякого конкретного j, перевірочному вузлу i не потрібно отримувати $L_{j\to i}$, оскільки його вплив все одно буде віднято.

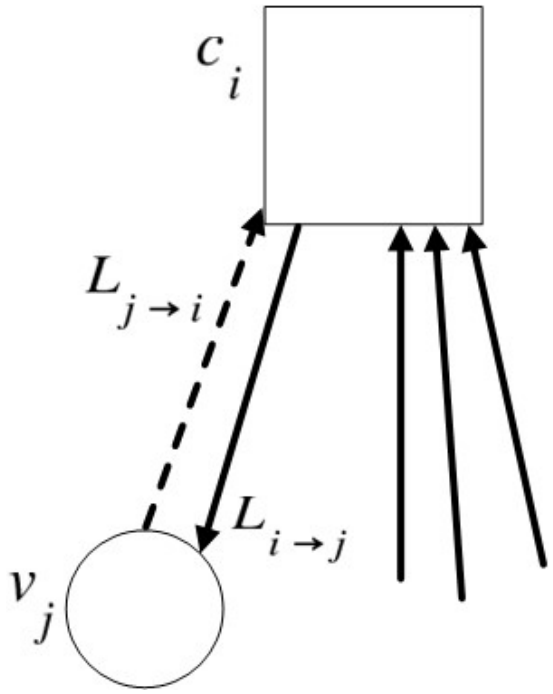

Рисунок 1.22 – Декодер перевірочних вузлів (декодер кодів з однією перевіркою на парність)

Декодери змінних та перевірочних вузлів працюють спільно та ітеративно для оцінки $L(v_j|y)$ для $j = 0, 1, \dots, n-1$. Будемо вважати, що застосовується план затоплення. Згідно з цим планом, всі змінні вузли обробляють свої вхідні дані та передають зовнішню інформацію до сусідніх перевірочних вузлів; перевірочні вузли обробляють свої вхідні дані і передають зовнішню інформацію до сусідніх змінних вузлів; і процедура повторюється, починаючи зі змінних вузлів. Після встановленої максимальної кількості повторень (або ітерацій) цього раунду декодування змінних та перевірочних вузлів або після того, як було виконано критерій зупинки, декодер обчислює (оцінює) КП $L(v_j|y)$, з якого приймається рішення щодо бітів v_j. Коли декодер має великі цикли – оцінки будуть дуже точними, а декодер матиме майже оптимальну продуктивність.

Розгляд алгоритму суми добутків спирається на припущення про незалежність: величина КП, отримана на кожному вузлі від своїх сусідів – незалежна. Очевидно, це припущення руйнується, коли кількість ітерацій перевищує половину обхвату графу Таннера.

Наразі на рисунку 1.21 та рисунку 1.22 було наведено концептуальні зображення функцій декодерів змінних та перевірочних вузлів, та було описано розподілений план декодування в контексті рисунку 1.20. На цьому етапі потрібно розглянути детальні операції в кожній складовому декодері (декодері змінних та перевірочних вузлів). Тобто трохи відступитися від розгляду декодера з максимальною апостеріорною імовірністю та процесорів апостеріорної

імовірності для кодів повторення і кодів з однією перевіркою на парність. Ці декодери є невід'ємною частиною загального LDPC декодера, який буде розглянуто пізніше.

Спершу слід розглянути більш детально декодер за максимумом апостеріорної імовірності для кодів повторення. Для цього розглянемо код повторення, в якому символ двійкового коду $c \in \{0, 1\}$ передається по каналу без пам'яті d разів, щоб отримати d-вектор r. За визначенням, декодер з максимальною апостеріорною імовірністю обчислює співвідношення логарифмічного КП

$$L(c|r) = \log\left(\frac{Pr(c = 0|r)}{Pr(c = 1|r)}\right), \tag{1.46}$$

або за однаково імовірного припущення для значення c, $L(c|r)$ є логарифмічним КП

$$L(c|r) = \log\left(\frac{Pr(r|c = 0)}{Pr(r|c = 1)}\right). \tag{1.47}$$

Це спрощується так:

$$L(c|r) = \log\left(\frac{\prod_{l=0}^{d-1} \Pr(r_l|c = 0)}{\prod_{l=0}^{d-1} \Pr(r_l|c = 1)}\right) = \sum_{l=0}^{d-1} \log\left(\frac{\Pr(r_l|c = 0)}{\Pr(r_l|c = 1)}\right) = \sum_{l=0}^{d-1} L(r_l|x)\,. \tag{1.48}$$

Таким чином, приймач за максимумом апостеріорної імовірності для коду повторення обчислює КП в каналі для кожного значення r_l та додає їх. Рішення за максимумом апостеріорної імовірності дорівнює $\hat{c} = 0$, якщо $L(c|r) \geq 0$ та $\hat{c} = 1$ в іншому випадку.

У контексті LDPC декодування, наведений вище вираз адаптований для обчислення зовнішньої інформації, що надсилається від змінного вузла j до перевірочного вузла i (рисунок 1.21),

$$L_{j\to i} = L_j + \sum_{i' \in N(j)-\{i\}} L_{i'\to j}\,, \tag{1.49}$$

де L_j – значення КП, обчислене на основі вибірки y_j з каналу.

$$L_j = L\big(c_j|y_j\big) = log\left(\frac{\Pr\big(c_j = 0\big|y_j\big)}{\Pr\big(c_j = 1\big|y_j\big)}\right). \tag{1.50}$$

Слід звернути уваги, що рівняння оновлення (1.49) відповідає формату (1.41). У контексті LDPC декодування змінні вузли називають процесором апостеріорної імовірності замість декодера за максимумом апостеріорної імовірності. На останній ітерації змінний вузол j приймає рішення на основі

$$L_j^{\text{рез}} = L_j + \sum_{i \in N(j)} L_{i\to j}\,. \tag{1.51}$$

Тепер розглянемо більш детально декодер за максимумом апостеріорної імовірності для кодів з однією перевіркою на парність. Для розгляду декодера з максимумом апостеріорної імовірності для кодів з однією перевіркою на парність спочатку звернемося до результатів Галагера.

Розглянемо вектор d незалежних двійкових випадкових величин $a = [a_0, a_1, \dots, a_{d-1}]$, в якому $\Pr(a_l = 1) = p_1^{(l)}$ та $\Pr(a_l = 0) = p_0^{(l)}$. Тоді імовірність того, що a містить парне число одиниць дорівнює

$$\frac{1}{2} + \frac{1}{2}\prod_{l=0}^{d-1}\left(1 - 2p_1^{(l)}\right), \tag{1.52}$$

та імовірність того, що a містить непарне число одиниць дорівнює

$$\frac{1}{2} - \frac{1}{2}\prod_{l=0}^{d-1}\left(1 - 2p_1^{(l)}\right). \tag{1.53}$$

Озброївшись цим результатом, тепер отримуємо декодер за максимальною апостеріорною імовірністю для кодів з однією перевіркою на парність. Розглянемо передачу кодового слова коду з однією перевіркою на парність довжини d через канал без пам'яті, вихід якого дорівнює r. Біти c_l в кодовому слові c мають єдине обмеження: у c має бути парне число одиниць. Не втрачаючи загальності, зосередимося на біті c_0, для якого правилом прийняття максимально правдоподібного рішення є

$$\hat{c}_0 = arg \max_{b\in\{0,1\}} \Pr(c_0 = b|r). \tag{1.54}$$

$$\Pr\{c_0 = 0|r\} = Pr\{c_1, c_2, \dots, c_{d-1} \text{ мають парну кількість одиниць } |r\} =$$

$$= \frac{1}{2} + \frac{1}{2}\prod_{l=1}^{d-1}(1 - 2\Pr(c_l = 1|r_l)), \tag{1.55}$$

де останній вираз слідує з (1.52). Перестановка дає

$$1 - 2\Pr(c_0 = 1|r) = \prod_{l=1}^{d-1}(1 - 2\Pr(c_l = 1|r_l)), \tag{1.56}$$

де використаємо той факт, що $p(c_0 = 0|r) = 1 - p(c_0 = 1|r)$. Можна замінити це на представлення КП, використовуючи відношення для загальної двійкової випадкової величини з імовірностями p_1 та p_0,

$$1 - 2p_1 = tanh\left(\frac{1}{2}log\left(\frac{p_0}{p_1}\right)\right) = tanh\left(\frac{1}{2}\text{КП}\right), \tag{1.57}$$

де $\text{КП} = \log(p_0/p_1)$. Застосовуючи це відношення до (1.56) отримуємо

$$\tanh\left(\frac{1}{2}L(c_0|r)\right) = \prod_{l=1}^{d-1} tanh\left(\frac{1}{2}L(c_l|r_l)\right), \tag{1.58}$$

або

$$L(c_0|r) = 2tanh^{-1}\left(\prod_{l=1}^{d-1} tanh\left(\frac{1}{2}L(c_l|r_l)\right)\right). \tag{1.59}$$

Таким чином, декодер максимальної правдоподібності для біта c_0 у коді з однією перевіркою на парність довжиною d приймає рішення $\hat{c}_0 = 0$, якщо $L(c_0|r) \geq 0$ та $\hat{c}_0 = 1$ в іншому випадку.

У контексті LDPC декодування, дотримуючись (1.59), коли перевірочні вузли працюють як процесори апостеріорної імовірності замість декодерів максимальної правдоподібності, перевірочний вузол i обчислює зовнішню інформацію

$$L_{i\to j} = 2tanh^{-1}\left(\prod_{j'\in N(i)-\{j\}} \tanh\left(\frac{1}{2}L_{j'\to i}\right)\right), \tag{1.60}$$

та передає її до змінного вузла j. Слід звернути увагу, що, оскільки, добуток перевищує множину $N(i)-\{j\}$ (рисунок 1.22), повідомлення $L_{j\to i}$ фактично віднімається для отримання зовнішньої інформації $L_{i\to j}$.

Рівняння (1.49) та (1.60) складають ядро декодера за алгоритмом суми добутків. Інформація $L_{j\to i}$, яку змінний вузол j надсилає перевірочному вузлу i на кожній ітерації, є найкращою (зовнішньою) оцінкою значення v_j (знаковий біт $L_{j\to i}$) та рівня довіри або надійності цієї оцінки (амплітуда $L_{j\to i}$). Ця інформація базується на обмеженні кодів повторення для змінного вузла j та всіх вхідних даних від сусідів змінного вузла j, за винятком перевірочного вузла i. Подібним чином, інформація $L_{i\to j}$, яку перевірочний вузол i надсилає змінному вузлу i на кожній ітерації, є найкращою (зовнішньою) оцінкою значення v_j (знаковий біт $L_{j\to i}$) та рівня впевненості або рівня надійності цієї оцінки (амплітуда $L_{j\to i}$). Ця інформація базується на обмеженні кодів з однієї перевіркою на парність для перевірочного вузла i та всіх вхідних даних від усіх сусідів перевірочного вузла i, за винятком змінного вузла j.

Поки рівняння (1.49) та рівняння (1.60) формують ядро декодера за алгоритмом суми добутків, декодеру все ще потрібен критерій зупинки ітерацій та крок ініціалізації. Стандартним критерієм припинення ітерацій є випадок, коли

$$\hat{v}H^T = 0 \tag{1.61}$$

де $\hat{v}$ – орієнтовне декодоване кодове слово.

Декодер ініціалізується шляхом встановлення всіх повідомлень змінних вузлів $L_{j\to i}$ рівними

$$L_j = L(v_j|y_j) = \log\left(\frac{Pr(v_j=0|y_j)}{Pr(v_j=1|y_j)}\right) \tag{1.62}$$

Для всіх j та i, для яких $h_{ij}=1$. Тут y_j представляє значення з каналу, яке було фактично отримане, тобто воно тут не є змінною.

Як було сказано на початку розділу, будуть розглянуті особливі випадки каналів. У випадку з двійковим каналом зі стираннями, коли $y_j \in \{0,1,e\}$ та визначимо $p = \Pr(y_j = e|v_j = b)$ як імовірність стирання, де $b \in \{0,1\}$, тоді легко переконатись, що

$$\left.\begin{array}{c}\Pr(v_j = b|y_j) = 1 \text{ коли } y_j = b,\\ \Pr(v_j = b|y_j) = 0 \text{ коли } y_j = b^c,\\ \Pr(v_j = b|y_j) = \frac{1}{2} \text{коли } y_j = e,\end{array}\right\} \quad (1.63)$$

де b^c – доповнення до b.

З цього випливає, що

$$\left.\begin{array}{c}L(v_j|y_j) = +\infty, y_j = 0,\\ L(v_j|y_j) = -\infty, y_j = 1,\\ L(v_j|y_j) = 0, y_j = e.\end{array}\right\} \quad (1.64)$$

У випадку двійкового симетричного каналу, $y_j \in \{0,1\}$ та визначимо $\varepsilon = \Pr(y_j = b^c|v_j = b)$ як імовірність помилки. Тоді очевидно, що

$$\left.\begin{array}{c}\Pr(v_j = b|y_j) = 1 - \varepsilon \text{ коли } y_j = b,\\ \Pr(v_j = b|y_j) = \varepsilon \text{ коли } y_j = b^c.\end{array}\right\} \quad (1.65)$$

Звідси, маємо

$$L(v_j|y_j) = (-1)^{y_j} \log\left(\frac{1-\varepsilon}{\varepsilon}\right). \quad (1.66)$$

Коли розглядаємо двійковий канал з адитивним білим Гаусовим шумом, спочатку нехай $x_j = (-1)^{v_j}$ буде j передане двійкове значення; слід звернути увагу на те, що $x_j = +1(-1)$ коли $v_j = 0(1)$. Далі використовуватимемо x_j та v_j як взаємозамінні. j отримане значення це $y_j = x_j + n_j$, де n_j є незалежним і зазвичай розподіляється як $\mathcal{N}(0, \sigma^2)$. Тоді легко показати

$$\Pr(x_j = x|y_j) = \left[1 + \exp\left(-\frac{2y_j x}{\sigma^2}\right)\right]^{-1}, \quad (1.67)$$

де $x \in \{\pm 1\}$, та звідси

$$L(v_j|y_j) = \frac{2y_j}{\sigma^2}. \quad (1.68)$$

На практиці, оцінка σ^2 є важливою.

У випадку незалежного каналу затухання Релея припускаємо, що декодер має ідеальну інформацію про стан каналу. Тут модель подібна до двійкового каналу з адитивним білим Гаусовим шумом: $y_j = \alpha_j x_j + n_j$, де $\{\alpha_j\}$ – незалежні випадкові величини Релея з єдиною дисперсією. Імовірність переходу в каналі в такому випадку становить

$$\Pr(x_j = x|y_j) = \left[1 + \exp\left(-\frac{2\alpha_j y_j x}{\sigma^2}\right)\right]^{-1}. \quad (1.69)$$

Тоді, змінні вузли ініціалізуються як

$$L(v_j|y_j) = \frac{2\alpha_j y_j}{\sigma^2}. \quad (1.70)$$

На практиці, оцінка σ^2 та α_j є важливою.

Отже, узагальнюючи усе вище написане, можна виписати покрокові інструкції як працює алгоритм декодування Галагера під назвою сума добутків.

По-перше, ініціалізація. Для усіх j треба ініціалізувати L_j згідно (1.62) для відповідної моделі каналу. Тоді для всіх i та j, для яких $h_{ij} = 1$, встановлюються $L_{j\to i} = L_j$.

По-друге, оновлення перевірочних вузлів. Обчислити вихідні повідомлення перевірочних вузлів $L_{i\to j}$ для кожного перевірочного вузла, використовуючи (1.60). І потім передати ці повідомлення до змінних вузлів. (Цей крок схематично показано на рисунку 1.22).

По-третє, оновлення змінних вузлів. Обчислити вихідні повідомлення змінних вузлів $L_{j\to i}$ для кожного змінного вузла, використовуючи (1.49). І потім передати ці повідомлення до перевірочних вузлів. (Цей крок схематично показано на рисунку 1.21).

По-четверте, обчислення результуючого коефіцієнту правдоподібності. Для $j = 0, 1, \dots, n-1$ обчислити $L_j^{\text{рез}}$ за (1.51).

У п'ятих, критерій призупинення ітерацій. Для $j = 0, 1, \dots, n-1$ встановити

$$\left.\begin{aligned} \hat{v}_j = 1 \text{ якщо } L_j^{\text{рез}} < 0, \\ \hat{v}_j = 0 \text{ в іншому випадку,} \end{aligned}\right\}$$

для того, щоб отримати $\hat{v}$. Якщо $\hat{v}H^T = 0$ або кількість виконаних ітерацій дорівнює встановленому максимуму – зупинити алгоритм; в іншому випадку повторити виконання з другого кроку.

Рівняння оновлення (1.60) для другого кроку є чисельно складним через наявність добутків та функцій $tanh$ та $tanh^{-1}$. За Галагером цю ситуацію можна покращити наступним чином. По-перше слід переписати коефіцієнт $L_{j\to i}$ на його знак та величину амплітуди (або значення біта та його надійність):

$$L_{j\to i} = \alpha_{ji}\beta_{ji}, \tag{1.71}$$

$$a_{ji} = sign\left(L_{j\to i}\right), \tag{1.72}$$

$$\beta_{ji} = \left|L_{j\to i}\right|. \tag{1.73}$$

Виходячи з цього (1.60) може бути переписано як

$$\tanh\left(\frac{1}{2}L_{i\to j}\right) = \prod_{j'\in N(i)-\{j\}} \alpha_{j'i} \cdot \prod_{j'\in N(i)-\{j\}} \tanh\left(\frac{1}{2}\beta_{j'i}\right). \tag{1.74}$$

Таким чином маємо

$$L_{i\to j} = \prod_{j'} \alpha_{j'i} \cdot 2\tanh^{-1}\left(\prod_{j'} \tanh\left(\frac{1}{2}\beta_{j'i}\right)\right) =$$

$$= \prod_{j'} \alpha_{j'i} \cdot 2\tanh^{-1}\log^{-1}\log\left(\prod_{j'} \tanh\left(\frac{1}{2}\beta_{j'i}\right)\right) =$$

$$= \prod_{j'} \alpha_{j'i} \cdot 2\tanh^{-1}\log^{-1}\sum_{j'} \log\left(tanh\left(\frac{1}{2}\beta_{j'i}\right)\right), \tag{1.75}$$

що дає нову форму для (1.60) у другому кроці,

$$L_{i\to j} = \prod_{j'\in N(i)-\{j\}} \alpha_{j'i} \cdot \phi\left(\sum_{j'\in N(i)-\{j\}} \phi(\beta_{j'i})\right), \tag{1.76}$$

де ми визначаємо

$$\phi(x) = -\log\left[\tanh\left(\frac{x}{2}\right)\right] = \log\left(\frac{e^x+1}{e^x-1}\right), \tag{1.77}$$

і використовуючи той факт, що $\phi^{-1}(x) = \phi(x)$ коли $x > 0$. Таким чином, рівняння (1.76) може бути використано замість (1.60) в другому кроці алгоритму декодування під назвою сума добутків. Функція $\phi(x)$ показана на рисунку 1.23 і може бути реалізована за допомогою пошукової таблиці. Однак, якщо необхідна дуже низька частота помилок, табличний декодер може страждати від частотних підлог помилок. Однією з можливих альтернатив табличному декодеру є лінійна апроксимація, як це обговорювалося в [21].

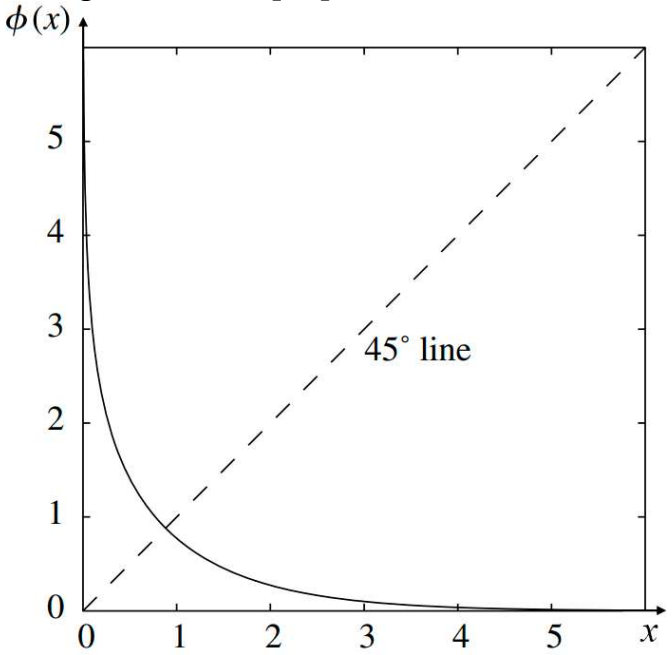

Рисунок 1.23 – Графік функції $\phi(x)$

1.4.2 Турбо-коди

Уперше турбо-коди були запропоновані в 1993 році К. Берроу, А. Главьє та П. Сітімашимой у [24]. У своїй роботі автори не тільки запропонували схему кодера, але й ефективний алгоритм декодування. Основними елементами схеми кодування є два рекурсивних систематичних згорктових кода, які пов’язані між собою перемежувачем, як показано на рисунку 1.24. З цієї причини турбо-коди також називають паралельними каскадними кодами.

В якості декодера було запропоновано ітеративну схему, ядром якої є два декодера згорткових кодів з м’яким виходом. Експериментально було доведено [24], що на довжині інформаційного слова 65536 біт така схема кодування та декодування практично досягає пропускну спроможність каналу при реалізованій складності декодера. В оригінальній роботі [24] для декодування

згорткового коду використовувався алгоритм під назвою Log-MAP [25], проте пізніше для зменшення складності декодування турбо-коду були розглянуті інші під оптимальні декодери з м'яким виходом, наприклад SOVA [26], Max-Log-MAP та Scaled Max-Log-MAP[27].

Незважаючи на погану мінімальну відстань, турбо-коди володіють гарними спектральними характеристиками та корегуючими властивостями, та практично одразу були оцінені по достоїнству. Багато робіт присвячено аналізу побудови турбо-кодів: досліджено вплив вибору перемежувача та згорткових кодів на характеристики коду [28], [29], [30], [31], розглянуті варіанти використання трьох та більше компонентних кодів [32]. Також значної уваги було приділено аналізу ітеративного декодера та способам його поліпшення: вивчені такі питання як швидкість сходження декодера [33], [34], [35], критерії зупинки декодера [36], [37] та схеми обміну зовнішніми повідомленнями імовірностей між компонентами декодера [38], [39].

Гарні характеристики турбо-кодів, легкість кодера та його гнучкість пояснюють їх популярність та широке практичне застосування. На даний момент турбо-коди застосовуються в системах стільникового зв'язку, таких як LTE, UMTS, CDMA2000, Wi-Max, в системах супутникового телемовлення DVB-RCS та супутникового зв'язку. З цієї причини дослідження кодів не зупиняються і на сьогоднішній момент.

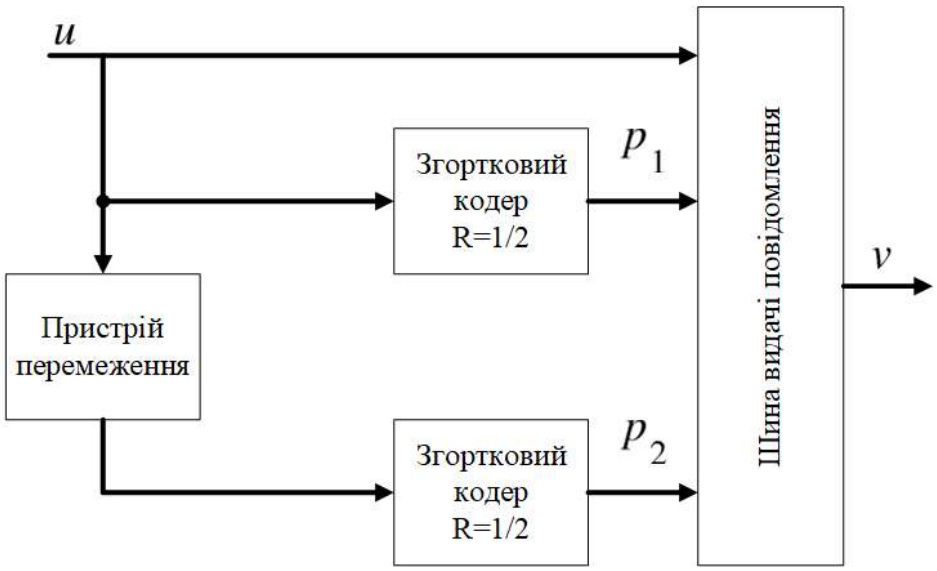

Рисунок 1.24 – Структурна схема кодера турбо-коду

Перед тим як перейти до розгляду турбо-коду слід описати властивості та параметри згорткових кодів, які є невід'ємною частиною турбо-коду. Також будуть розглянуті алгоритми декодування згорткових кодів з жорстким та м'яким виходом, які використовуються при декодуванні турбокодів.

1.4.2.1 Згорткові коди

Згорткові коди – це напів-нескінченні лінійні коди. На відміну від блокових кодів, котрі працюють з блоками даних однакової довжини, в загальному випадку згортковий код обробляє інформаційне слово довільної довжини, а

виходом кодера є напів-нескінченна послідовність закодованих значень. Зазвичай кажуть, що швидкість згорткового коду дорівнює $R = b/c$, якщо на кожен блок з b вхідних біт, кодер генерує c вихідних значень. В такому випадку вхідна і вихідна послідовності відповідно позначаються як $u = \{u_0, u_1, u_2, \dots\}$ та $v = \{v_0, v_1, v_2, \dots\}$, де $u_i = \left\{u_i^{(0)}, \dots, u_i^{(b-1)}\right\}$ та $v_i = \left\{v_i^{(0)}, \dots, v_i^{(c-1)}\right\}$.

Кодер згорткового коду може бути представлено як лінійний регістр зсуву. Розрізняють рекурсивні та нерекурсивні згорткові коди, які можуть бути представлені у вигляді лінійного регістру зсуву зі зворотнім зв'язком та без зворотного зв'язку, відповідно. Вихідні значення нерекурсивного коду залежать лише від інформаційних бітів, які зберігаються в комірках пам'яті, в той час як в рекурсивному коді враховуються вихідні біти.

В загальному вигляді згортковий код зі швидкістю b/c виглядає як b регістрів зсуву, пов'язаних між собою суматорами по модулю 2. Якщо позначити кількість комірок пам'яті як v_i в кожному з них, то сумарне значення $v = \sum_{i=1}^{b} v_i$ називається кодовим обмеженням цього коду, а величина $\max_i\{v_i\}$ – пам'яттю коду. Таким чином, складність кодера визначається кількістю комірок пам'яті в регістрах та кількістю суматорів по модулю 2. Приклад згорткового коду з регістром зсуву для системного коду без зворотного зв'язку зі швидкістю $1/2$ показано на рисунку 1.25.

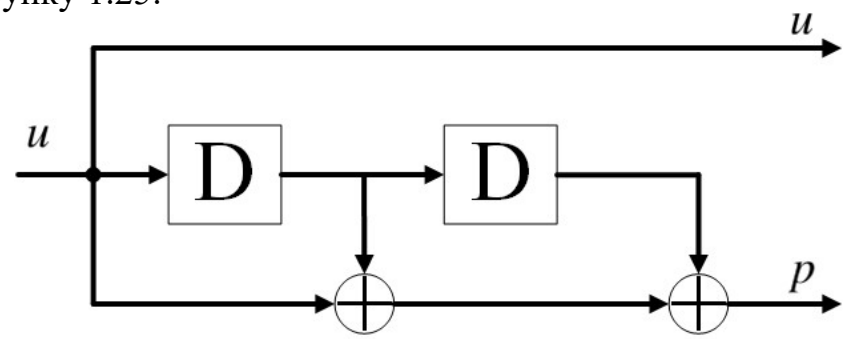

Рисунок 1.25 – Згортковий кодер зі швидкістю $1/2$

Згортковий код також можна задати в поліноміальному вигляді. В такому випадку вхідну та вихідну послідовності записують у вигляді багаточленів

$$u(x) = \left[u^{(0)}(x), u^{(2)}(x), \dots, u^{(b-1)}(x)\right], \tag{1.78}$$

та

$$v(x) = \left[v^{(0)}(x), v^{(2)}(x), \dots, v^{(c-1)}(x)\right]. \tag{1.79}$$

Зобразимо ряд $u^{(j)}(x) = u_0^{(j)} + u_1^{(j)}x + u_2^{(j)}x^2 + u_3^{(j)}x^3 + \cdots$ та аналогічно для $v^{(j)}(x)$.

Процес кодування можна представити як добуток полінома, який відповідає вхідній послідовності, з генераторною матрицею багаточленів

$$u(x) = v(x)G(x). \tag{1.80}$$

Для отримання генераторної матриці багаточленів, кожен регістр з відповідними зв'язками також записується у вигляді багаточлена. В загальному випадку його може бути представлено як

$$g(x) = \frac{f(x)}{q(x)} = \frac{f_0 + f_1 x + f_2 x^2 + \cdots + f_{v_i} x^{v_i}}{1 + q_1 x + q_2 x^2 + \cdots + q_{v_i} x^{v_i}}, \quad (1.81)$$

де f_i – коефіцієнт прямого зв'язку в регістрі,

q_i – коефіцієнт зворотного зв'язку.

Значення i коефіцієнту дорівнює одиниці, якщо в регістрі є відвід від i комірки пам'яті, в протилежному випадку, коефіцієнт дорівнює нулю. У випадку відсутності зворотного зв'язку в регістрі $q(x) = 1$. Використовуючи поліноміальний запис регістрів, результуюча матриця буде виглядати як

$$G(x) = \begin{pmatrix} g_{11}(1) & g_{12}(x) & \dots & g_{1c}(x) \\ g_{11}(1) & g_{12}(x) & \dots & g_{1c}(x) \\ \vdots & \vdots & \ddots & \vdots \\ g_{11}(1) & g_{12}(x) & \dots & g_{1c}(x) \end{pmatrix}. \quad (1.82)$$

Для регістру, який показано на рисунку 1.25, поліноміальна генераторна матриця дорівнює $G(x) = [1, 1 + x + x^2]$. З поліноміальної форми матрицю можна переписати в двійковому вигляді. Тоді генераторна матриця коду буде мати напів-нескінчений блоково-діагольний вигляд, і для коду $[1, 1 + x + x^2]$ може бути записана таким чином

$$G = \begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & \dots & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & \dots & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & \dots & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & \dots & 0 \\ & & & & & & & \dots & & & & & & \end{bmatrix}. \quad (1.83)$$

Кількість можливих станів регістру зсуву залежить від кодового обмеження коду і дорівнює $N = 2^v$. Так як вихідні значення кодера залежать лише від вхідних бітів і від поточного стану регістру, то увесь процес кодування можна представити як роботу кінцевого автомату, який може бути представлено у вигляді діаграми переходів. З кожного стану такого кінцевого автомату виходить 2^b ребер, на яких відмічені значення відповідних вхідних і вихідних бітів. Приклад такої діаграми для коду, який розглядається наведено на рисунку 1.26.

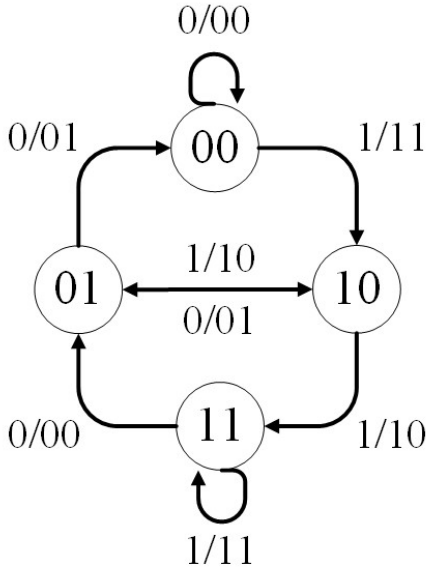

Рисунок 1.26 – Діаграма станів згорткового коду

Якщо діаграму розгорнути у часі, то буде отримана так звана гратчаста діаграма, яку приведено на рисунку 1.27.

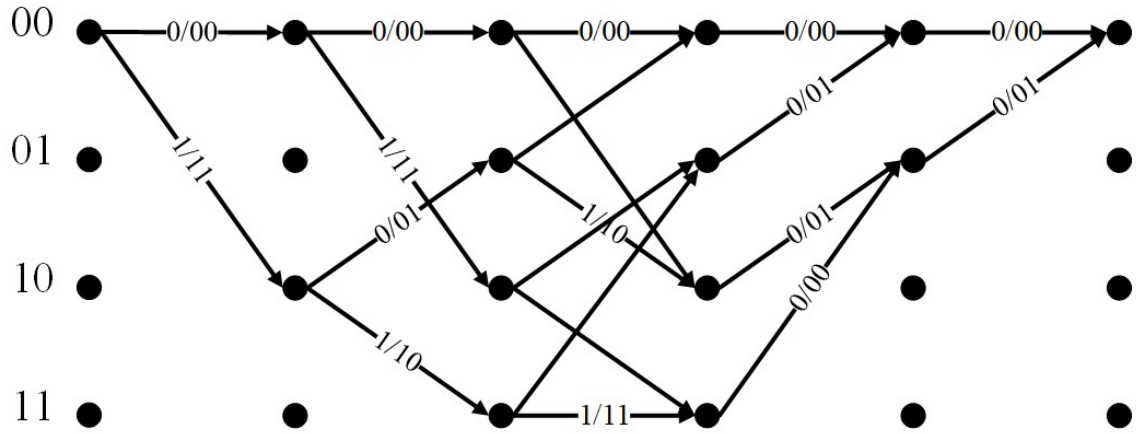

Рисунок 1.27 – Гратчаста діаграма згорткового коду

Кожній інформаційній послідовності відповідає шлях або набір станів на гратчастій діаграмі. Представлення згорткового коду у вигляді гратчастої діаграми дуже зручне під час декодування повідомлень. В такому випадку, задача декодера зводиться до знаходження найбільш імовірного шляху на діаграмі.

Як і для блокових кодів, під час аналізу корегуючої здатності згорткового коду розглядають його мінімальну відстань. Але на відміну від блокових кодів, згортковий код характеризується множиною мінімальних відстаней. Так як відстань послідовності на вході кодера може бути довільною, то для кодових послідовностей різної довжини відстань буде теж різною

$$d_k = \min\{d(v, v')\}, \tag{1.84}$$

де v – кодова послідовність довжини k;

v' – кодова послідовность довжини k;

d – відстань Хеммінга.

Довжина k називається шириною вікна декодування, та чим більше його значення, тим більше помилок може скорегувати код. Якщо розглядати кодові послідовності на гратчастій діаграмі, то очевидно, що зі збільшенням кодової послідовності мінімальна відстань не зменшується, тобто

$$d_1 \le d_2 \le d_3 \le \cdots \le d_\infty. \tag{1.85}$$

Проте відстань коду не збільшується нескінченно зі збільшенням k і в якийсь момент досягає насичення. Найбільше значення мінімальної відстані називається вільною відстанню коду і позначається як

$$d_{free} = \max\{d_1, d_2, d_3, \dots, d_\infty\}. \tag{1.86}$$

Іншою важливою характеристикою згорткового коду є спектр вільних відстаней коду, який визначає кількість кодових послідовностей, які мають певне значення вільної відстані. В той час як для блокових кодів існують конструктивні методи знаходження гарних кодів, для знаходження гарних згорткових кодів використовують комп’ютерний пошук. Даний факт обмежує можливість використання довгих згорткових кодів, бо пошук є трудомісткою задачею.

Оскільки при використанні згорткових кодів в реальних системах зв'язку вхідна послідовність ділиться на блоки певної довжини, постає завдання коректного закінчення кодування. Найпростішим підходом є використання усіченої кодової конструкції. В даному випадку шлях на діаграмі згорткового коду починається в нульовому стані, тобто початкове значення пам'яті регістра дорівнює нулю, та може закінчуватися довільним значенням. Недолік цього методу у тому, що під час декодування невідомо кінцевий стан шляху на діаграмі, що призводить до менш надійного декодування кінцевих бітів інформаційного слова. Саме тому розглядають дві альтернативні конструкції: додавання нульового хвоста (термінована конструкція) або циклічне замикання (кільцева конструкція або тейлбайтинг). В термінованій конструкції до інформаційного слова додаються v бітів, які призводять шлях на діаграмі до нульового стану. Але в такому випадку доводиться передавати додаткові біті. Кільцева конструкція позбавлена цього недоліка. При використанні циклічного замикання початкові значення в комірках пам'яті згорткового коду ініціалізуються таким чином, щоб початкові та кінцеві стани шляху на діаграмі співпадали. Наприклад, для нерекурсивних кодів перед кодуванням в комірки регістру записуються v останніх бітів інформаційної послідовності, тоді при завершенні кодування останніх v бітів інформаційного слова початкові та кінцеві стани регістра будуть однаковими. Дана властивість дозволяє надійно декодувати останні біти інформаційного слова і при цьому не передавати додаткові біти.

З практичної точки зору найбільший інтерес викликають згорткові коді зі швидкістю $1/c$. Це викликано тим, що механізм перфорації (або виколювання) бітів дозволяє із згорткового коду зі швидкістю $1/c$ отримати код з довільною більшою швидкістю. Наприклад, при перфорації кожного c біту з кодовою послідовності можна отримати код зі швидкістю $1/(c-1)$. При цьому отримані таким чином високошвидкісні коди можуть бути представлені за допомогою однієї і тієї ж самої гратчастої діаграми, а отже декодовані за допомогою одного і того ж алгоритму. Це дозволяє мати код з різними швидкостями, але який буде декодуватися одним декодером.

1.4.2.2 Декодування згорткових кодів

У цій роботі основну увагу приділено систематичним згортковим кодам зі швидкістю $1/c$, тож при розгляді алгоритмів декодування передбачається їх використання. Проте слід відмітити, що усі викладки справедливі і для несистематичних кодів.

Перед тим як перейти до розгляду введемо наступні позначення. Нехай інформаційна послідовність довжиною k і має вигляд:

$$u = [u_0, u_1, \dots, u_{k-1}].$$

Тоді, вихідна послідовність кодера

$$v = [v_0, v_1, \dots, v_{k-1}],$$
$$v_i = \left[v_i^{(0)}, v_i^{(1)}, \dots, v_i^{(c-1)}\right].$$

Кодові біти, які відповідають інформаційному біту u_i. Також позначимо сигнальне відображення бітів кодового слова як $a_i = \left[a_i^{(0)}, a_i^{(1)}, \dots, a_i^{(c-1)}\right]$, де $a_i^j = \sqrt{E_s}\left(2v_i^{(j)} - 1\right)$. Тобто, якщо $v_i^{(j)} = 1$, то $a_i^{(j)} = \sqrt{E_s}$, в іншому випадку $\mathrm{a}_i^{(j)} = -\sqrt{E_s}$. Як E_s позначена енергія, яка витрачається для передачі одного біта кодового слова, котра співвідноситься з енергією на біт інформаційного слова E_b як $E_s = RE_b$.

Структурна схема системи передачі даних, яка розглядається показана на рисунку 1.28.

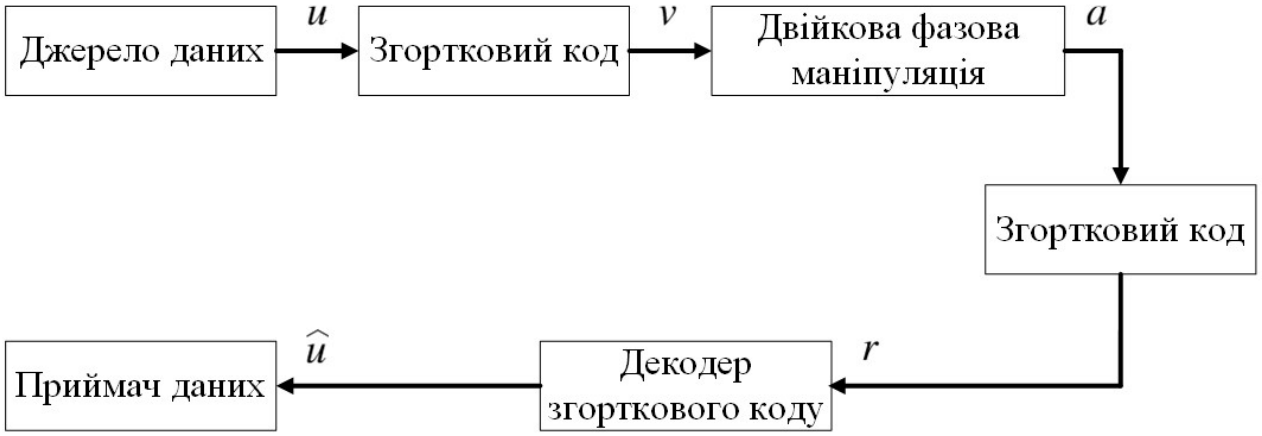

Рисунок 1.28 – Структурна схема системи передачі даних

Зазвичай в якості каналу розглядають канал з адитивним білим Гаусовим шумом. Це пов'язано з тим, що перемежувач кодованих бітів, який враховує характер помилок в каналі, переводить будь-які групи помилок, котрі були викликані поміхами в каналі, в незалежні. Під час передачі через такий канал вихідні значення дорівнюють $r_i = \left[r_i^{(0)}, r_i^{(1)}, \dots, r_i^{(c-1)}\right]$, де $r_i^{\,j} = a_i^{(j)} + n_i^{(j)}$, а $n_i^{(j)}$ незалежна Гаусова величина з нульовим математичним очікуванням і спектральною щільністю потужності шуму дорівнюючій $N_0/2$. Для такої величини щільність умовної імовірності $Pr(r_i^{(j)}|v_i^{(j)})$ дорівнює

$$\Pr\left(r_i^{(j)}\middle|v_i^{(j)}\right) = \frac{1}{\sqrt{2\pi}\sigma}\exp\left(-\frac{1}{2\sigma^2}\left(r_i^{(j)} - a_i^{(j)}\right)^2\right), \qquad (1.87)$$

де σ – дисперсія шуму.

Так як канал з адитивним білим Гаусовим шумом є каналом без пам'яті, то справедливий наступний вираз

$$\Pr(r_i|v_i) = \frac{1}{\left(\sqrt{2\pi}\sigma\right)^c}\exp\left(-\frac{1}{2\sigma^2}\|r_i - a_i\|^2\right), \qquad (1.88)$$

де $\|\cdot\|^2$ – Евклідова відстань.

Під час передачі через канал без пам'яті задача декодування коду полягає у знаходженні послідовності станів в гратчастій діаграмі згорткового коду, яка може бути представлена як Марківський процес. Оптимальними декодерами згорткових кодів, які основані на їх гратчастом представленні є алгоритм Вітербі та BCJR. Декодер Вітербі може працювати як з жорсткими, так і з м'якими

входами, в той час як BCJR є декодером з м'яким входом. Декодер Вітербі є алгоритмом, який мінімізує імовірність помилки на слово і обирає декодоване слово u таким чином, щоб максимізувати імовірність $\Pr(r|u)$, тобто є декодером за максимумом правдоподібності.

$$\hat{u} = \arg\max_{u} \Pr(r|u), \tag{1.89}$$

де $\hat{u}$ – прийняте рішення про декодоване слово.

Алгоритм BCJR вирішує іншу задачу і мінімізує імовірність помилки на біт. Алгоритм є декодером за максимумом апостеріорної імовірності і максимізує імовірність $\Pr(u_i|r)$

$$\hat{u}_i = \arg\max_{\hat{u}_i} \Pr(u_i|r), \tag{1.90}$$

де $\hat{u}_i$ – прийняте рішення про декодований біт.

На відміну від алгоритму Вітербі BCJR обчислює імовірність вихідних бітів декодованого інформаційного слова $\Pr(\hat{u}_i|r)$. Але для алгоритму Вітербі існує розширення, яке дозволяє розраховувати приблизні імовірності декодованих бітів, називається це розширення SOVA (англ. – Soft Output Viterbi Algorithm – SOVA).

М'які рішення декодера зручно представляти у вигляді логарифму відношення правдоподібності, або надійністю бітів

$$L(\hat{u}_i) = \log\left(\frac{\Pr(\hat{u}_i = 1|r)}{\Pr(\hat{u}_i = 0|r)}\right), \tag{1.91}$$

де log – натуральний логарифм.

У випадку, коли $L(\hat{u}_i) > 0$, то $\hat{u}_i = 1$, в іншому випадку декодер приймає рішення, що $\hat{u}_i = 0$. Для того, щоб із значення надійності знову перейти у імовірнісну область треба скористатися властивістю, що

$$\Pr(\hat{u}_i = 1|r) + \Pr(\hat{u}_i = 0|r) = 1. \tag{1.92}$$

Тоді значення $\Pr(\hat{u}_i = 1|r)$ та $\Pr(\hat{u}_i = 0|r)$ можна обчислити наступним чином

$$\Pr(\hat{u}_i = 1|r) = \frac{1}{1 + \exp\left(-L(\hat{u}_i)\right)}, \tag{1.93}$$

$$\Pr(\hat{u}_i = 0|r) = \frac{\exp\left(-L(\hat{u}_i)\right)}{1 + \exp\left(-L(\hat{u}_i)\right)}. \tag{1.94}$$

Входом декодерів зазвичай також є не імовірності, а надійності $L\left(r_i^{(j)}\right) = log\frac{\Pr\left(r_i^{(j)}\middle|v_i^{(j)} = 1\right)}{\Pr\left(r_i^{(j)}\middle|v_i^{(j)} = 0\right)}$. Для каналу з адитивним білим Гаусовим шумом

$$L\left(r_i^{(j)}\right) = L_c r_i^{(j)}. \tag{1.95}$$

Складова виразу (1.95) має вигляд

$$L_c = \frac{2\sqrt{E_s}}{\sigma^2}.$$

Для переходу в імовірнісну область також можна скористатися формулами (1.93) та (1.94).

Алгоритм Вітербі було запропоновано в 1967 році А. Вітербі в [40], проте його теперішня інтерпретація та доказ оптимальності були зроблені Д. Форні в [41]. Як вже було сказано, алгоритм Вітербі мінімізує імовірність помилки на слово, що еквівалентно максимізації $\Pr(u|r)$. Використовуючи правило Байєса, цю імовірність можна виразити як

$$\Pr(u|r) = \frac{\Pr(u|r)\,\Pr(u)}{\Pr(r)}. \tag{1.96}$$

Значення $\Pr(r)$ однакове для усіх можливих u та його можна опустити тому, що воно є нормуючим коефіцієнтом. Також в багатьох випадках значення u_i вважають рівноімовірними, що призводить до того, що задача максимізації $\Pr(u|r)$ зводиться до задачі максимізації $\Pr(r|u)$, яку вирішує декодер Вітербі. Проте у випадку, коли u_i не рівноімовірні, $\Pr(u)$ необхідно враховувати.

Так як u та v однозначно задають шлях на діаграмі, то $\Pr(r|u) = \Pr(r|v)$ і для каналу без пам'яті може бути обчислена як

$$\Pr(r|v) = \prod_{i=0}^{k-1} \log(Pr(r_i|v_i)). \tag{1.97}$$

Стосовно до згорткових кодів – $\log \Pr(r|v)$ називається метрикою шляху і позначається як $M(r|v)$, а $\log \Pr(r_i|v_i)$ називається метрикою ребра та позначається як $M(r_i|v_i)$. Для каналу з адитивним білим Гаусовим шумом метрика ребра виводиться з вираження (1.88) та має наступний вигляд

$$M(r_i|v_i) = -c \log\left(\sqrt{2\pi}\sigma\right) - \frac{1}{2\sigma^2}\|r_i - a_i\|^2. \tag{1.98}$$

Розглянемо секцію діаграми під номером i. У випадку двійкового згорткового коду зі швидкістю $1/c$ в кожен стан діаграми входить два ребра, які відповідають вхідним інформаційним бітам 0 і 1. Назвемо частковою метрику того шляху, який починається в нульовій секції діаграми і закінчується в деякому стані секції i. Позначимо таку часткову метрику для двох шляхів, що розглядаються, які в секції $i-1$ закінчуються у стані p_1 та p_2, як показано на рисунку 1.29, відповідно як $M_{i-1}(p_1)$ та $M_{i-1}(p_2)$. Тоді метрика шляхів, які переходять із станів p_1 та p_2 в стан q, може бути обчислена як

$$M_i(q) = M_{i-1}(p_1) + M\left(r_i, \hat{v}_i^{(p_1,q)}\right), \tag{1.99}$$

або в іншому випадку

$$M_i(q) = M_{i-1}(p_2) + M\left(r_i, \hat{v}_i^{(p_2,q)}\right), \tag{1.100}$$

де $M\left(r_i, \hat{v}_i^{(p,q)}\right)$ – метрика переходу (p, q).

Метриці відповідають деякі вихідні кодові біти, позначені як $\hat{v}_i^{(p,q)}$.

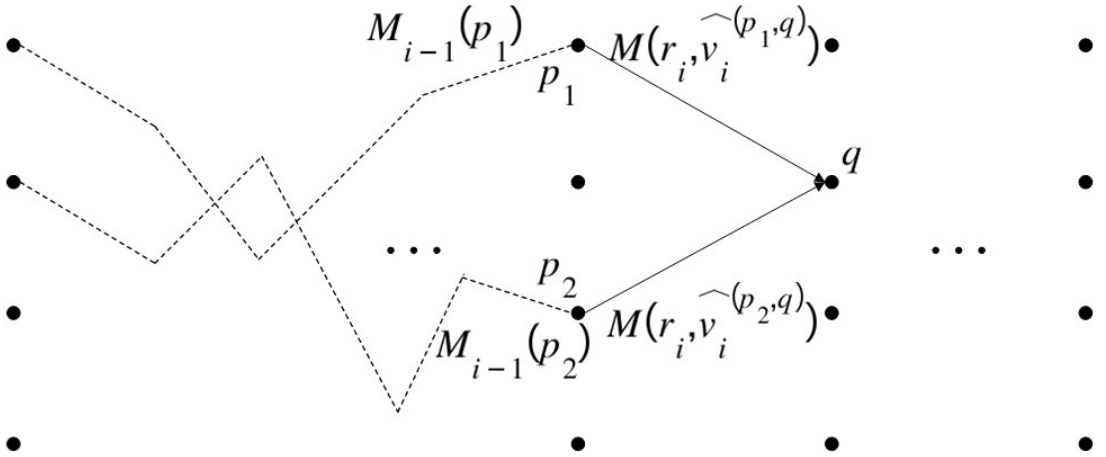

Рисунок 1.29 – Приклад роботи декодера Вітербі на діаграмі згорткового коду

Основою метою алгоритму Вітербі є максимізація сумарної метрики шляху. Тому даний алгоритм керується принципом оптимальності Беллмана, який для даного випадку можна сформулювати наступним чином: «Для знаходження найкоротшого шляху на діаграмі, шлях, який входить в стан q повинен бути найкоротшим. Інакше можна буде знайти більш коротший шлях на діаграмі» [42]. Таким чином, основним правилом алгоритму Вітербі є вибір шляху з максимальним значенням часткової метрики в кожному стані діаграми. Такий шлях називається тим, що вижив, а його метрика буде обчислена як

$$M_i(q) = \max\left\{M_{i-1}(p_1) + M\left(r_i, \hat{v}_i^{(p_1,q)}\right), M_{i-1}(p_2) + M\left(r_i, \hat{v}_i^{(p_2,q)}\right)\right\}. \quad (1.101)$$

Так як до кінця діаграми невідомо який із шляхів виживе та буде найкращим, то в кожному стані діаграми потрібно зберігати часткову метрику шляху, що вижив, та номер стану, з якого прийшов цей шлях. Коли ці дії будуть виконані для останньої секції в діаграмі, то в результаті буде знайдено найкращий шлях. Для того, щоб відновити інформаційне слово, необхідно виконати зворотній прохід по діаграмі. Оскільки в кожному стані було збережено номер стану із котрого прийшов найкращий шлях, то це може бути легко прорoблено. Також слід врахувати, що складність алгоритму має експоненціальну залежність від значення кодового обмеження, так як обчислення метрик шляхів та вибір найкращого шляху відбувається в кожному стані діаграми.

В загальному випадку алгоритм Вітербі описується наступним чином:

1. якщо відомо початковий стан шляху на діаграмі згорткового коду, то встановити $i = 0, M_0(0) = 0$ та $M_0(j) = -\infty$, для усіх $j = 1, \dots, N-1$. Якщо початковий стан невідомий, то усі $M_0(j) = 0$, для усіх $j \in \{0, \dots, N-1\}$;
2. збільшити i на одиницю та обчислити метрики переходів;
3. для кожного стану q знайти метрики шляхів, що входять за формулами (1.99) та (1.100);
4. обрати шлях з максимальною величиною метрики для стану q у відповідності з виразом (1.101);

5. зберегти стан p, з якого прийшов шлях з максимальною частковою метрикою;
6. повторити кроки з другого по шостий для усіх станів діаграми;
7. якщо відомо, що діаграма замикається в нульовому стані, то із кінцевого стану виконати зворотній прохід по діаграмі найкращим шляхом та знайти інформаційне слово;
8. якщо кінцевий стан шляху на діаграмі невідомо, то знайти кінцевий стан з максимальною метрикою шляху та з нього виконати зворотній прохід по діаграмі для відновлення інформаційного слова.

Під час роботи алгоритму Вітербі в каналі з адитивним білим Гаусовим шумом в логарифмічній області зручніше користуватися іншою метрикою шляху і ребра, а саме кореляцією між прийнятою послідовністю та декодованими бітами. Тоді метрика переходу секції i, якому відповідають деякі кодові біти $\left[\hat{v}_i^0, \hat{v}_i^1, \ldots, \hat{v}_i^{(c-1)}\right]$, обчислюється як

$$M_{corr}(r_i, v_i) = \sum_{j=1}^{c} r_i^{(j)} \hat{v}_i^{(j)} . \qquad (1.102)$$

Часткові метрики шляхів обчислюються так само, як показано вище.

У 1989 році Хагенауер та Хоехер [26] запропонували алгоритм Вітербі з м'яким виходом (SOVA), який приблизно розраховує надійності декодованих бітів. Основні кроки роботи алгоритму повторюють алгоритм Вітербі з невеликими відмінностями. Як і в алгоритмі Вітербі, SOVA виконує прямий і зворотній проходи по діаграмі для находження декодованого біту. Проте, крім цього, виконується ще k часткових проходів у зворотному напрямку, в протягом котрих обчислюються надійності бітів. В кожному стані алгоритм SOVA окрім інформації про найкращий шлях зберігає різницю часткових метрик між шляхом, що вижив і тим, що було відкинуто

$$\Delta_i^q = \max\left(M_{i-1}(p_1) + M\left(r_i, \hat{v}_i^{(p_1,q)}\right), M_{i-1}(p_2) + M\left(r_i, \hat{v}_i^{(p_2,q)}\right)\right) -$$
$$- \min\left(M_{i-1}(p_1) + M\left(r_i, \hat{v}_i^{(p_1,q)}\right), M_{i-1}(p_2) + M\left(r_i, \hat{v}_i^{(p_2,q)}\right)\right). \qquad (1.103)$$

В роботі [26] показано, що різниці Δ_i^q відповідає надійність коректності вибору шляху, що вижив. А так як шлях, що вижив та відкинутий шлях на деякій частині мають різні відповідні інформаційні бітові послідовності, то ця метрика може бути використана для обчислення надійності останніх.

Нехай початкові та кінцеві стани на діаграмі дорівнюють нулю, тоді всі шляхи на діаграмі мають загальне початковий та кінцевий стани. Розглянемо всі шляхи на діаграмі, які використовувалися для обчислення Δ_i^q та позначимо як $l = i - \delta$ секцію діаграми на якій вони розійшлися. Звичайні значення δ не перевищують п'яти – шести кодових обмежень коду. На дільниці от l до i секцій шляхи, що розглядаються мають різні переходи і, відповідно, різні інформаційні бітові послідовності. Якщо в секції i вибір найкращого шляху було зроблено правильно, то на позиціях де біти двох шляхів різняться, станеться помилка, імовірність якої буде залежати від значення Δ_i^q. Так як в кожному стані

найкращого шляху алгоритм SOVA відкидає шлях з найменшою метрикою та надійність кожного біту залежить від ряду відкинутих шляхів, то результуючі значення надійності біт, які різняться у найкращого шляху та відкинутого в стані q, будуть обчислені як

$$L(\hat{u}_j) = \min\{L(\hat{u}_j), \Delta_i^q\}, \quad (1.104)$$

для усіх $j \in \{l, \dots, l+\delta\}$, де біти різняться і де початкове значення надійності біту дорівнює $L(\hat{u}_j) = \infty$. Знак надійності задається значенням біту $\hat{u}_j$, тобто знак негативний, якщо $\hat{u}_j = 0$, і позитивний, якщо $\hat{u}_j = 1$. Таким чином, алгоритм SOVA описується наступними кроками:

1. виконання алгоритму Вітербі для знаходження шляхуз максимальною метрикою та збереження різниці Δ_i^q для кожного із станів найкращого шляху за формулою (1.103);
2. встановлення надійності кожному біту, яка б дорівнювала $L(\hat{u}_j) = \infty$;
3. для обчислення надійності вихідних бітів, спочатку з кожного стану найкращого шляху виконати зворотній прохід уздовж відкинутого шляху. Слідом, для бітів, що різняться в найкращого шляху та відкинутого шляху, обчислити надійність біту за формулою (1.104);
4. знак надійності виставити у відповідності зі значенням декодованого біту.

Для корегування значень надійності на виході алгоритму SOVA в роботі [43] було запропоновано подальший спосіб покращення. В основі алгоритму лежить той факт, що запускаючи алгоритм SOVA на початку діаграми та з кінця, декодовані слова будуть однаковими, але розраховані значення надійності будуть різними. Таким чином, можна отримати дві послідовності надійності бітів, починаючи декодування з початку та з кінця діаграми. В якості результуючої надійності біту обирається мінімальне значення з двох отриманих послідовностей. Не дивлячись на те, що такий підхід дозволяє більш точно розрахувати вихідні надійності біт, він рідко застосовується на практиці.

Алгоритм BCJR було запропоновано Л. Балем та іншими в роботі [25] як алгоритм, який працює за максимумом апостеріорної імовірності для знаходження імовірностей станів та бітів в Марківських ланцюгах з кінцевою кількістю станів. Так як вихід згорткового коду, який пройшов через канал без пам’яті можна рахувати Марківським джерелом, то можливе використання даного алгоритму для його декодування. Алгоритм BCJR обчислює апостеріорні імовірності кожного переходу по діаграмі, які у подальшому використовуються для обчислення апостеріорної імовірності бітів.

Поділимо вихід каналу на три частини:

- $r_{<i}$ – символи, які відповідають секціям до моменту часу i;
- r_i – символи, які відповідають секції діаграми з номером i;
- $r_{>i}$ – символи, які відповідають майбутнім секціям діаграми.

Розглянемо секцію діаграми згорткового коду в момент часу i (рисунок 1.30), де можна побачити перехід зі стану $S_i = p$ в стан $S_{i+1} = q$. Апостеріорна

імовірність переходу (p, q), з урахуванням того, що $r = (r_{<i}, r_i, r_{>i})$ може бути виражена як

$$\Pr(S_i = p, S_{i+1} = q|r) = \frac{p(S_i = p, S_{i+1} = q, r_{<i}, r_i, r_{>i})}{p(r)}. \quad (1.105)$$

Даний вираз можна переписати в розгорнутому вигляді, використовуючи той факт, що джерело Марківське

$$\Pr(S_i = p, S_{i+1} = q|r) = \frac{p(S_i = p, r_{<i})p(S_{i+1} = q, r_i|S_i = p)p(r_{>i}|S_{i+1} = q)}{p(r)}. \quad (1.106)$$

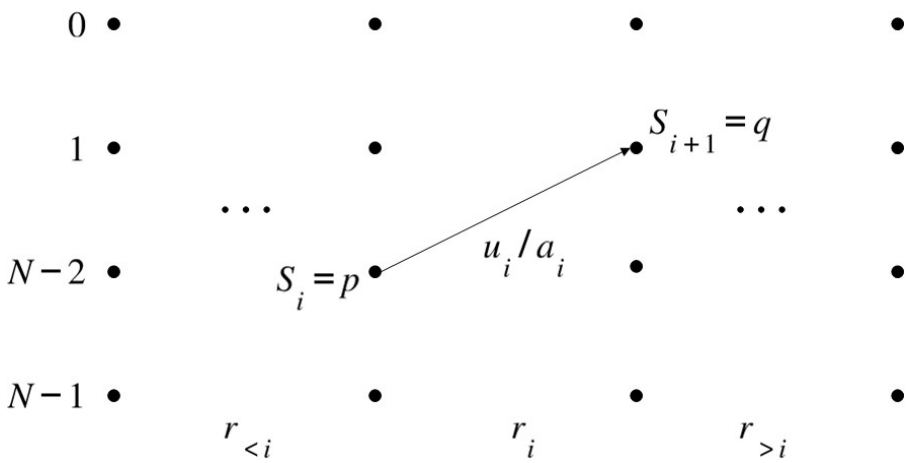

Рисунок 1.30 – Перехід (p, q) в діаграмі згорткового коду

Перше значення у чисельнику позує імовірність того, що шлях на діаграмі закінчиться у стані p в момент часу i з урахуванням прийнятої послідовності $r_{<i}$. Другий вираз – це імовірність того, що шлях на діаграмі проходить через стани (p, q) з урахуванням прийнятого значення r_i. Третій вираз – це імовірність того, що шлях на діаграмі починається в стані q в момент часу i з урахуванням прийнятих з каналу символів $r_{>i}$. Зазвичай ці імовірності позначають так

$$\alpha_i(p) = p(S_i = p, r_{<i}), \quad (1.107)$$

$$\gamma_i(p, q) = p(S_{i+1} = q, r_i|S_i = p), \quad (1.108)$$

$$\beta_{i+1}(q) = p(r_{>i}|S_{i+1} = q). \quad (1.109)$$

Тоді вираз для знаходження апостеріорної імовірності може бути представлено в більш лаконічному вигляді

$$\Pr(S_i = p, S_{i+1} = q|r) = \frac{\alpha_i(p)\gamma_i(p, q)\beta_{i+1}(q)}{p(r)}. \quad (1.110)$$

Перед тим як перейти до методу розрахунку величин (1.107), (1.108) та (1.109) припустимо, що для секції i діаграми розраховані всі апостеріорні імовірності для всіх переходів. Розділимо всі переходи в секції i на дві підмножини. Ті, котрі були ініційовано одиничним інформаційним бітом, позначимо як Q_1, а ті, котрі відповідають нульовому інформаційному біту, позначимо як Q_0. Тоді апостеріорна імовірність біту буде обчислена наступним чином

$$\Pr(\hat{u}_i = x|r) = \sum_{(p,q)\in Q_x} \Pr(S_i = p, S_{i+1} = q|r) =$$

$$= \frac{1}{p(r)} \sum_{(p,q)\in Q_x} \alpha_i(p)\gamma_i(p,q)\beta_{i+1}(q)\,, \qquad (1.111)$$

для усіх значень $x \in \{0,1\}$.

Для обчислення значень $\alpha_{i+1}(q)$ та $\beta_i(p)$ використовуються рекурентні вирази. Припустимо, що значення $\alpha_i(p)$ відомі для усіх станів $p \in \{0,\dots,N-1\}$ в секції i діаграми, тоді $\alpha_{i+1}(q)$ може бути обчислено як

$$\alpha_{i+1}(q) = \sum_{q=0}^{N-1} \alpha_i(p)\gamma_i(p,q)\,. \qquad (1.112)$$

Значення $\beta_i(p)$ обчислюються за тим же принципом. Відмінність полягає у тому, що обчислення для $\alpha_{i+1}(q)$ починаються спочатку діаграми, в той час як для $\beta_i(p)$ обчислення проводиться з її кінця. Припустимо, що значення $\beta_{i+1}(q)$ для всіх станів $q \in \{0,\dots,N-1\}$ відомі, тоді $\beta_i(p)$ може бути обчислено як

$$\beta_i(p) = \sum_{q=0}^{N-1} \beta_{i+1}(q)\gamma_i(p,q)\,. \qquad (1.113)$$

Процес обчислення значень $\alpha_{i+1}(q)$ та $\beta_i(p)$ показано на рисунку 1.31.

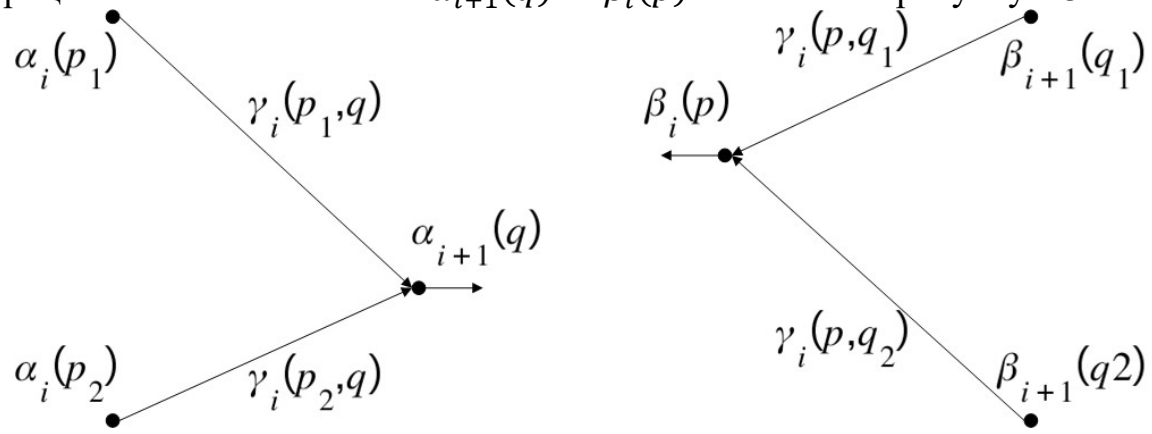

Рисунок 1.31 – Обчислення значень $\alpha_{i+1}(q)$ та $\beta_i(p)$

Початкові значення $\alpha_0(0)$ та $\beta_k(0)$ залежать від згорткового коду. Якщо відомо, що початкове та кінцеве значення стану шляху на діаграмі згорткового коду дорівнюють нулю, то $\alpha_0(0) = 1$ і $\beta_k(0) = 1$, в той час, як для решти станів початкові значення дорівнюють нулю. Якщо кінцевий та початковий стан кодера невідомий, то при декодуванні всі кінцеві та початкові стани вважаються рівноімовірними та дорівнюють $1/N$.

Розрахунок значень α та β для усіх станів діаграми виконується в ході прямого та зворотного проходу по діаграмі згорткового коду. Як видно з виразів (1.112) та (1.113) невідомою величиною залишається імовірність переходу (p,q), тобто $\gamma_i(p,q)$. Спосіб обчислення значень $\gamma_i(p,q)$ залежить від значень, які були прийняті з каналу та від типу каналу. Врахуємо те, що перехід (p,q) в секції i діаграми однозначно задається вхідним бітом $u^{(p,q)}$ і однозначно задає вихідні значення $v^{(p,q)}$. Тоді для коду зі швидкістю $R = 1/c$ і каналу з адитивним білим Гасовим шумом значення $\gamma_i(p,q)$ може бути обчислено як

$$\gamma_i(p,q) = \frac{1}{\left(\sqrt{2\pi}\sigma\right)^c} \exp\left(-\frac{1}{2\sigma^2}\left\|r_i - \hat{a}_i^{(p,q)}\right\|^2\right) \Pr(u_i = x), \qquad (1.114)$$

де $\Pr(u_i = x)$ – апріорна імовірність того, що u_i дорівнює $x \in \{0,1\}$.

Зазвичай апріорна імовірність дорівнює $1/2$, проте у випадку турбо-декодування це не так, і саме ця імовірність передається від одного декодера до іншого.

Таким чином, алгоритм BCJR зводиться до наступних кроків:

1. якщо відомо, що початковий та кінцевий стан в діаграмі дорівнює нулю, то встановити $\alpha_0(0) = 1$ і $\beta_k(0) = 1$, а $\alpha_0(q) = 0$ і $\beta_k(q) = 0$ для усіх $q \in \{1, \dots, N-1\}$. В іншому випадку $\alpha_0(q) = 1/N$ і $\beta_k(q) = 1/N$ для усіх $q \in \{0, \dots, N-1\}$;
2. обчислити метрики переходів $\gamma_i(p,q)$ для усіх ребер за формулою (1.114);
3. обчислити метрики станів $\alpha_i(p)$ в ході прямого проходу по діаграмі за формулою (1.112);
4. обчислити метрики станів $\beta_{i+1}(q)$ в ході зворотного проходу по діаграмі за формулою (1.113);
5. обчислити апостеріорні імовірності вихідних бітів за формулою (1.111).

1.4.2.3 Турбо кодер

Як можна побачити на рисунку 1.24, кодер турбокоду складається з двох рекурсивних систематичних згорткових кодів (англ. – Recursive Systematic Convolutional Codes – RSC codes) зі швидкістю $R = 1/2$, які пов'язані між собою k-бітним перемежувачем. Згорткові коди, які використовуються в процесі кодування турбокоду часто називають компонентними кодами. Вибір саме рекурсивних кодів пов'язано з тим, що вага кодового слова RSC коду сильно залежить від розташування одиниць в інформаційному слові, на відміну від нерекурсивного, і він забезпечує більшу вагу вихідного слова при малій вазі вхідного. В свою чергу турбокоди, які використовують RSC коди, мають більшу мінімальну відстань та кращу корегуючу здатність.

В оригінальній роботі [24] турбокод має швидкість $R = 1/3$, а його кодово слово виглядає як $v = (u, p_1, p_2)$, де p_1 та p_2 – перевірочні біти першого та другого згорткових кодів відповідно. В загальному вигляді можливе використання більшої кількості компонентних кодів, але принцип побудови коду не зміниться. Компонентні коди можуть бути як різними, так і однаковими, проте досвід позує, що використання різних компонентних кодів не дає переваг.

Перед тим як поступити на вхід другого згорткового коду, інформаційні біти проходять через перемежувач. Припустимо, що вага Хеммінга послідовності u дорівнює t, тоді отримана на виході перемежувача послідовність $\Pi(u)$ також буде мати вагу t. Функція $\Pi(u)$ задає перемежувач. Проте перевірочні біти на виході кожного компонентного коду будуть різними і мати різну вагу. Отже, якщо одна із перевірочних послідовностей буде мати малу вагу, можна розраховувати на те, що друга перевірочна послідовність буде мати більшу ваги

і загальна відстань коду буде більше. Також слід відмітити, що корельовані послідовності на входах компонентних декодерів негативно впливають на ефективність турбо-декодера в цілому. Перемежувач декорелює вхідні послідовності та дозволяє використовувати ітеративний підхід. Таким чином, перемежувач вирішує дві задачі: дозволяє створювати коди з великою відстанню та дає можливість використовувати ефективний ітеративний декодер.

Довжина перемежувача в значній мірі впливає на ефективність коду і чим вона більше, тим краще код [44]. Існує декілька видів перемежувачів. Найбільш простим з практичної точки зору є блоковий перемежувач, який представляється у вигляді матриці. Для перестановки вхідні біти записуються в матрицю по рядках і вичитуються по стовбцях. Також значення можуть бути вичитані по діагоналі. Інший вид перемежувачів називається випадковим. Щоб задати його, генерується n чисел у випадковому порядку і у відповідності з отриманою послідовністю відбувається перестановка бітів. За тим же принципом працює S-випадковий перемежувач, для якого послідовність цілих чисел генерується за спеціальними правилами [45]. В роботі [31] сказано, що для коротких слів краще використовувати невипадкові перемежувачі, в той час як для великих довжин випадкові переважніше.

Як і в разі згорткових кодів, для турбокодів можливе використання механізму виколювання бітів для отримання коду з більшою швидкістю. Після виколювання швидкість коду обчислюється як $R = k/(k + y)$, де y – це кількість перевірочних бітів, які залишилися після виколювання. При цьому структура декодера не змінюється, що є важливою якістю та завдяки чому ці коди мають широке практичне застосування.

Як було сказано вище, турбокоди мають погану мінімальну відстань d_{min}, проте мають гарні корегуючі здатності. Це пов’язано з тим, що кодових слів з вагою d_{min} в турбокоді мало і основна їх доля припадає на слова більшої ваги. Тому в турбокодах велику роль відіграє середня відстань коду та розподіл ваги кодових слів. З цієї причини загальною рисою, яка характеризує всі турбокоди, є різкий спад графіку залежності імовірності помилки на біт BER від відношення сигнал/шум, після чого швидкість спаду різко зменшується. Дана область називається областю насичення імовірності помилки або «полицею». Одним із способів зменшення висоти полиці є вибір гарного перемежувача [44].

Так як довжина перемежувача кінцева і визначає довжина вхідного інформаційного слова, то можна визначити вагу кожного кодового слова d, кількість слів з такою вагою N_d та сумарну вагу Хеммінга відповідних їм інформаційних слів w_d. Адитивна межа імовірності помилки BER турбокоду може бути отримана так

$$BER \leq \sum_{d=d_{free}}^{\infty} \frac{w_d}{k} Q\left(\sqrt{\frac{2RdE_b}{N_0}}\right), \qquad (1.115)$$

де d_{free} – вільна відстань коду;

k – довжина інформаційного слова, біт.

Часто на практиці необхідно оцінювати висоту області насичення імовірності помилки. Це може бути зроблено за допомогою моделювання, але якщо дана область оцінюється, наприклад, для $BER \leq 10^{-9}$, то моделювання стає занадто довгим. Знаючи ці значення та враховуючи те, що полиця обумовлена кодовими словами малої ваги, можливо оцінити висоту полиці за допомогою наступного виразу [46]

$$BER \approx \frac{w_{free}}{k} Q\left(\sqrt{\frac{2Rd_{free}E_b}{N_0}}\right), \tag{1.116}$$

де w_{free} – сумарна вага інформаційних слів, яким відповідають кодові слова з вагою d_{free}.

Для отримання оцінки роботи ітеративного турбо-декодера зазвичай до обчисленого таким чином значення додається 0,5 дБ для корекції, що пов'язано з підоматимальністю ітеративного декодера [46]. Але отримані аналітичні оцінки достатньо погано передбачають поведінку коду в області спаду імповірності помилки, і вони не можуть бути використані для порівняння турбокодів між собою. Для цього використовується моделювання.

1.4.2.4 Декодування турбокоду

Розглянемо ітеративну схема декодування, яку було запропоновано в [24]. Як було сказано вище, ядром схеми є два декодера згорткових кодів з м'яким виходом. Представимо прийняте з каналу слово як $r = (r_s, r_{p1}, r_{p2})$, де r_s відповідає прийнятій систематичній частині кодового слова, а r_{p1} та r_{p2} – його перевірочні символи. Тоді (r_s, r_{p1}) та $(\Pi(r_s), r_{p2})$ є входом першого і другого компонентного декодера відповідно. В своїй роботі К. Берроу та інші [24] показали, що для систематичного згорткового коду вихідні значення $L(\hat{u}_i)$ алгоритму декодування, можуть бути представлені як

$$L(\hat{u}_i) = L_{sys}(\hat{u}_i) + L_{app}(\hat{u}_i) + L_{ext}(\hat{u}_i), \tag{1.117}$$

де $L_{sys}(\hat{u}_i)$ – систематична складова надійності;

$L_{app}(\hat{u}_i)$ – апріорна складова надійності;

$L_{ext}(\hat{u}_i)$ – зовнішня складова надійності.

Обчислення апріорної надійності для наступного декодера проходить за правилом, яке слідує з формули (1.117):

$$L_{ext}(\hat{u}_i) = L(\hat{u}_i) - L_{sys}(\hat{u}_i) - L_{app}(\hat{u}_i). \tag{1.118}$$

Апріорна або внутрішня складова – це інформація, яку відомо про біти ще до декодування. У випадку ітеративного декодування турбокоду апріорні імовірності один декодер отримує від іншого. Систематична або апостеріорна складова надійності біту – це інформація, яка отримана з каналу, тобто $L_{sys}(\hat{u}_i) = L\left(r_s^{(i)}\right)$, де $r_s^{(i)}$ – це i символ, який відповідає систематичній частині прийнятого кодового слова. Зовнішню інформацію $L_{ext}(\hat{u}_i)$ декодер обчислює виходячи із прийнятих з каналу значень і апріорної інформації, але без урахування $r_s^{(i)}$ і відповідної i символу апріорної інформації. Саме ця складова є

апріорною для наступного декодера. Детальну схему ітеративного декодера з турбо-принципом для декодування компонентних кодів показано на рисунку 1.32.

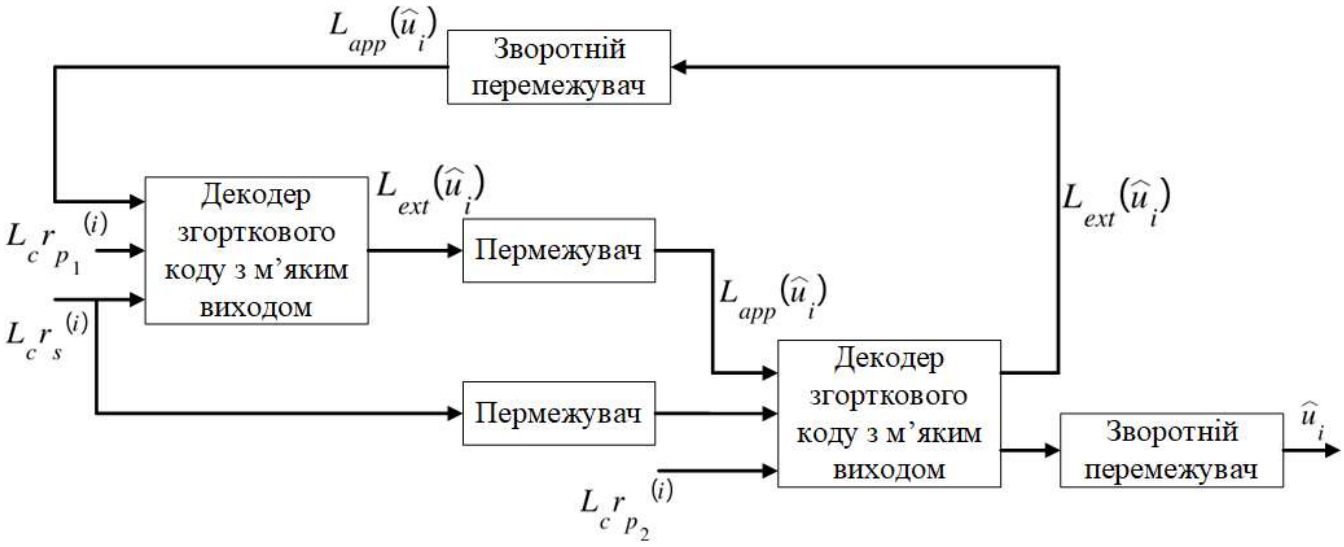

Рисунок 1.32 – Структурна схема ітеративного турбо декодера

На початку кожного декодування апріорні надійності на вході першого декодера дорівнюють нулю. В свою чергу перший компонентний декодер обчислює зовнішні надійності, які інтерпретуються як апріорні на вході другого декодера на наступній пів-ітерації. Другий декодер використовує отриману інформацію для більш надійного декодування бітів та обчислює нову послідовність зовнішніх значень надійності для подальшого ітерування. Таким чином, ітеративно змінюються і корегуються апріорні надійності бітів, що дозволяє більш точно декодувати прийняте слово.

В залежності від використовуваного алгоритму, який було обрано для декодування компонентного коду, а також параметрів каналу, кількість ітерацій декодера може різнитися. В момент, коли подальше ітерування декодера не приводить до зміни знаку вихідних значень надійності бітів, говорять, що декодер зійшовся. В роботах [33], [47] показано, що турбо-декодер, який використовує механізм обміну зовнішніми імовірностями, завжди сходиться до деякого стаціонарного стану та закінчує свою роботу. Зазвичай досить 8-16 ітерацій для того, щоб декодер зійшовся [44]. Для оцінки того, зійшовся декодер чи ні, і для дострокового завершення ітерування користуються різними критеріями зупинки [36], [37], [48]. Їх можна розділити на три групи:

1. критерії, що використовують м'які рішення компонентних декодерів. До них можна віднести взаємну ентропію зовнішніх значень надійності, частоту зміни останніх на різних ітераціях, мінімальне та середнє обчислених значень надійності [48];
2. критерії, які використовують жорсткі рішення компонентних декодерів. В даному випадку враховуються декодовані послідовності на різних ітераціях [36];
3. критерії, що основані на додаткових знаннях про інформаційне слово. Часто на практиці для перевірки коректності декодування

використовують CRC, яка також може бути застосована для дострокової зупинки турбо-декодера [37].

Окремо розглядають способи прискорення сходження ітеративного декодера для зменшення затримку декодування. На схемі, яку показано на рисунку 1.33, компонентні декодери послідовно обмінюються зовнішніми значеннями надійності, проте можливі й інші схеми обміну. В роботі [38] розглянуто альтернативна схема обміну. У випадку паралельного обміну два компонентних декодера працюють одночасно, після чого обмінюються зовнішніми імовірностями. Посимвольна схема є покращеною паралельною схемою і пропонує негайно передавати знову обчислену зовнішню імовірність кожного компонентного декодера. Таким чином, в обох підходах кожен декодер згорткового коду може використовувати більш актуальну інформацію для підрахунку м'яких рішень. Як наслідок, такі декодери дозволяють зменшити загальну кількість необхідних ітерацій для декодування, проте виграшу за BER практично не дають [38], [49]. На рисунку 1.33 показано принцип роботи двох підходів обміну імовірностями.

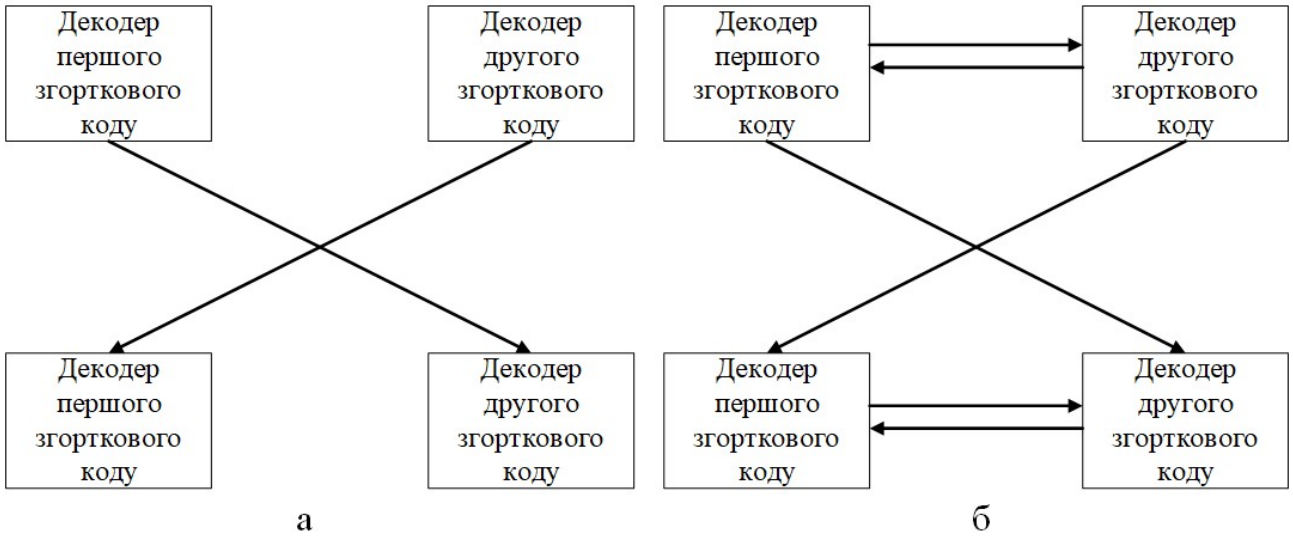

Рисунок 1.33 – Схеми обміну зовнішніми значення надійності:
а – паралельна схема;
б – посимвольна схема

Як слідує із сказаного вище, для декодування турбокоду може бути використано будь-який декодер згорткового коду з м'яким виходом. В деяких алгоритмів кращі показники сходження, а деякі мають кращі корегуючи здатності.

2 МАТЕМАТИЧНЕ МОДЕЛЮВАННЯ ТА ПОРІВНЯЛЬНИЙ АНАЛІЗ РАДІОЛІНІЙ З РІЗНИМИ МЕТОДАМИ ЗАВАДОЗАХИЩЕННОГО КОДУВАННЯ

2.1 Математичне моделювання LDPC-кодування

У випадку радіоліній з космічними апаратами швидкість передачі даних є відносно низькою, що дозволяє кодам низької швидкості вміщуватися у виділеному діапазоні, незважаючи навіть на те, що розширення смуги пропускання більше. Сигнали у радіолініях зв'язку з космічними апаратами проходять значні відстані, що вимагає надзвичайно гарних показників відношення сигнал/шум. Це досягається використанням кодів, які мають велику надмірність (низьку швидкість) і більшу складність (тому для досягнення найкращої продуктивності їм може знадобитися більше ітерацій). Оскільки швидкість передачі даних нижча, функції, необхідні для кращої роботи, не є тягарем для системи. З цих причин для використання у радіолініях зв'язку з космічними апаратами було рекомендовано набір LDPC кодів, що належать до сімейства AR4JA (англ. – Accumulate, Repeat-by-4, and Jagged Accumulate – AR4JA) [50].

Обрані коди з сімейства мають швидкості $1/2$ та $4/5$. А довжини інформаційних блоків кодів з цього сімейства складають $k = 1024$ та $k = 4096$. Обране сімейство є систематичним кодом.

Для математичного моделювання LDPC-кодування було використано модель каналу на рисунку 1.1.

Генераторні та перевірочні матриці обраного сімейства кодів показано на рисунку 2.1 – рисунку 2.8. Чорні точки на рисунках відображають одиниці в матрицях коду.

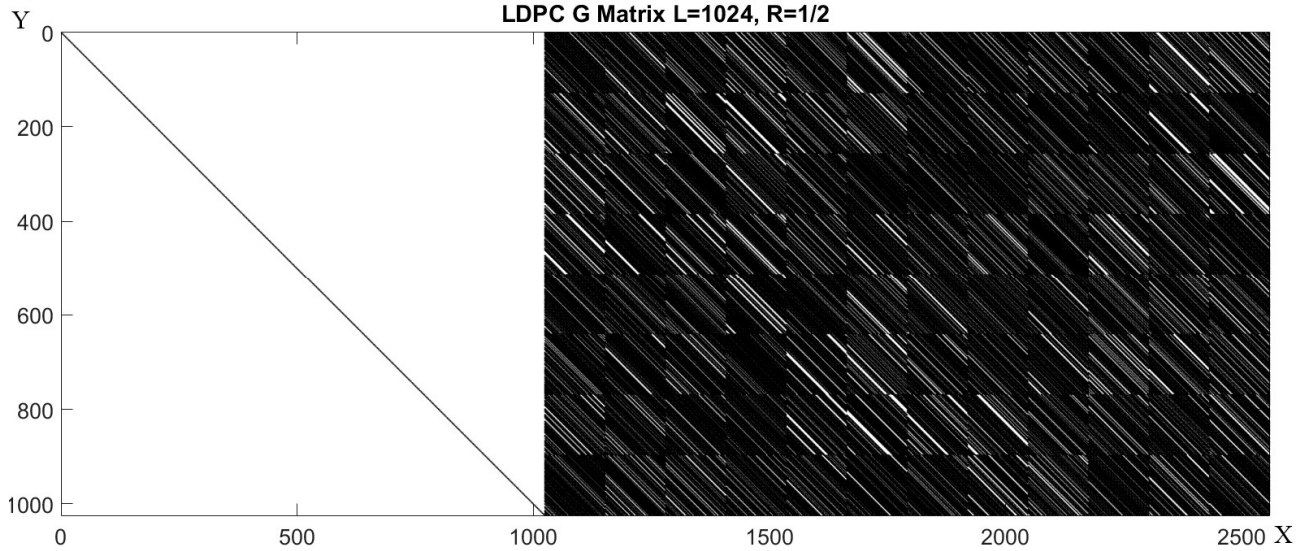

Рисунок 2.1 – Зображення генераторної матриці G LDPC коду для швидкості $1/2$ та довжини інформаційного слова 1024 біти

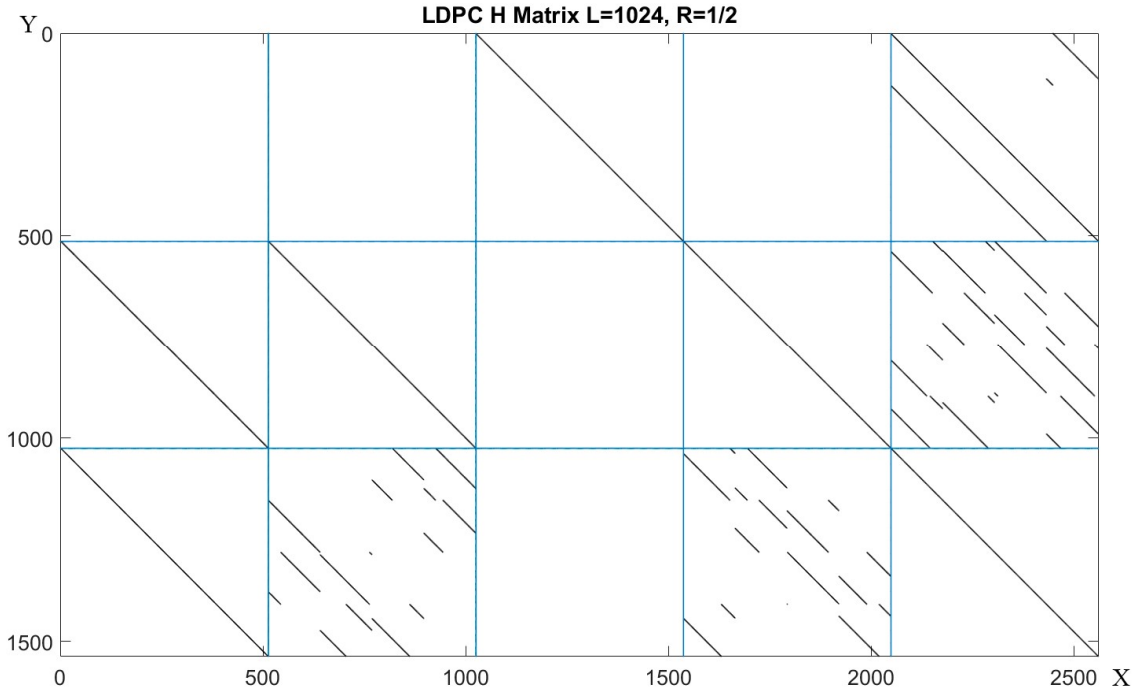

Рисунок 2.2 – Зображення перевірочної матриці H LDPC коду для швидкості $1/2$ та довжини інформаційного слова 1024 біти

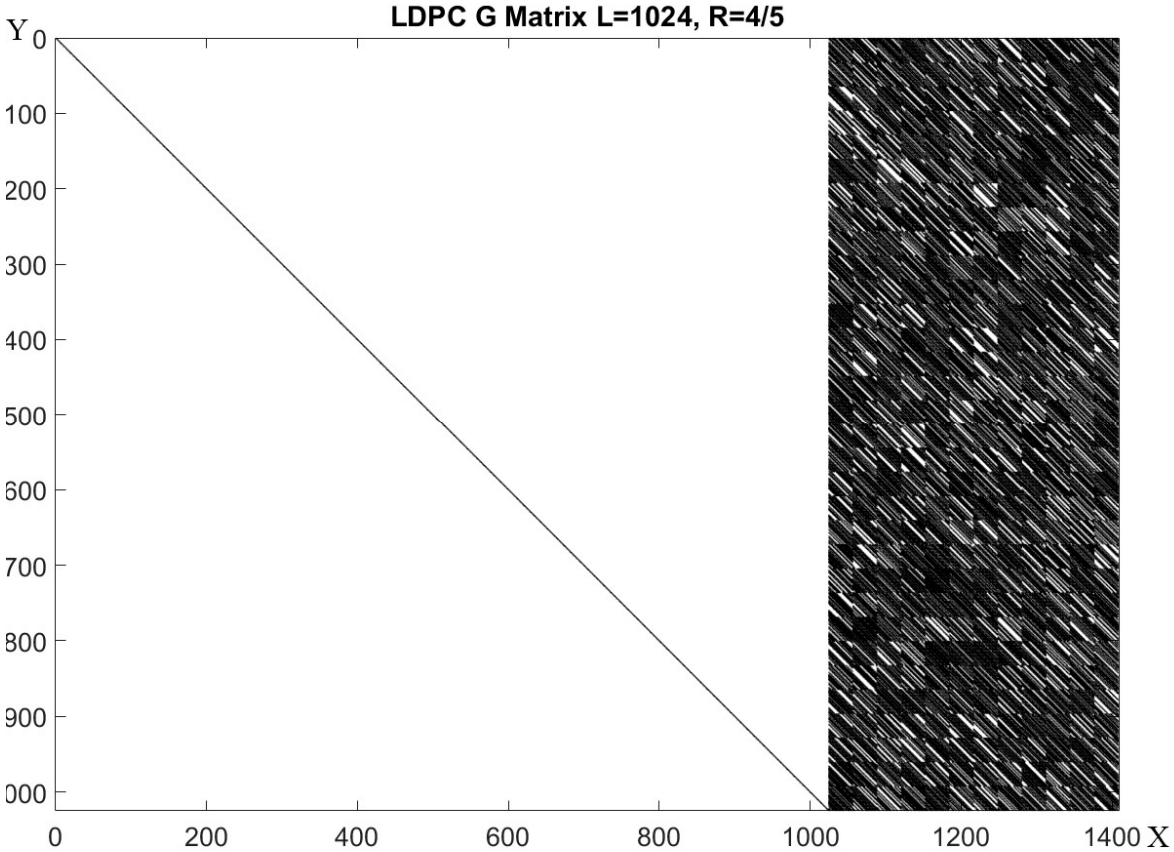

Рисунок 2.3 – Зображення генераторної матриці G LDPC коду для швидкості $4/5$ та довжини інформаційного слова 1024 біти

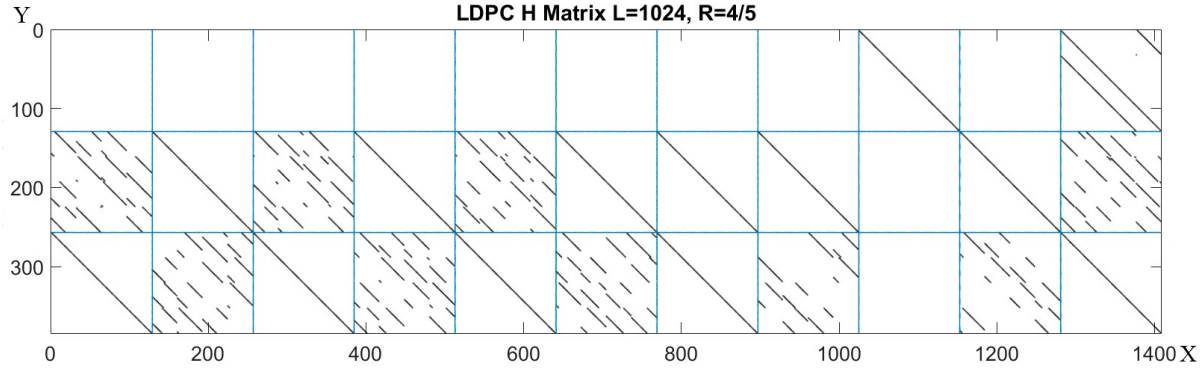

Рисунок 2.4 – Зображення перевірочної матриці H LDPC коду для швидкості 4/5 та довжини інформаційного слова 1024 біти

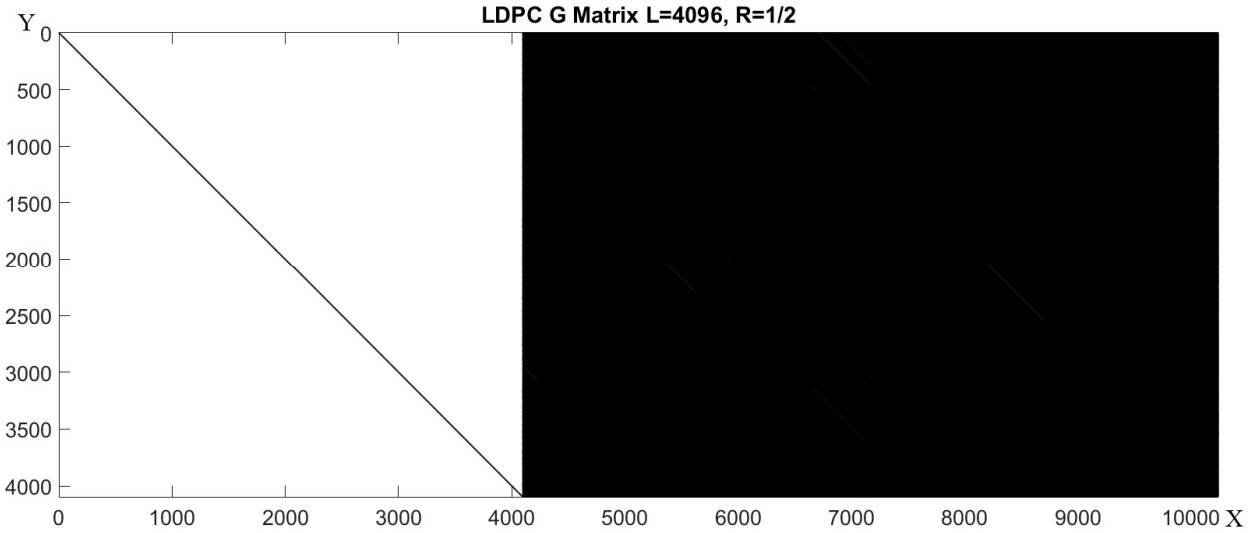

Рисунок 2.5 – Зображення генераторної матриці G LDPC коду для швидкості 1/2 та довжини інформаційного слова 4096 бітів

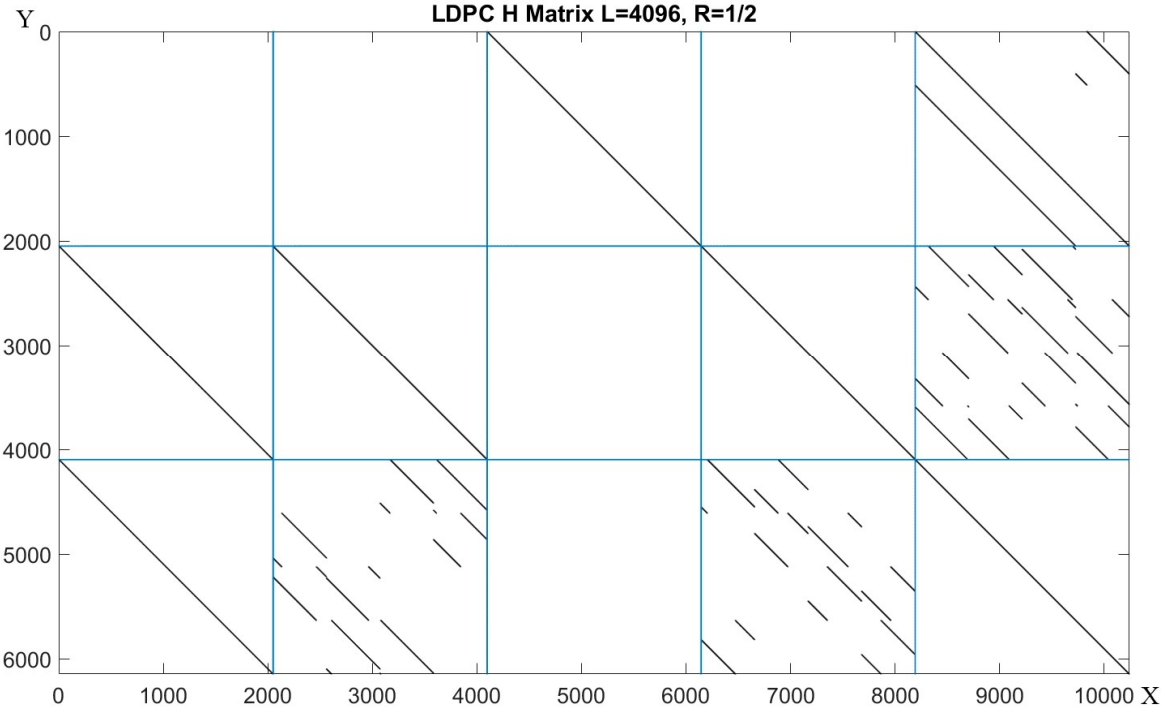

Рисунок 2.6 – Зображення генераторної матриці H LDPC коду для швидкості 1/2 та довжини інформаційного слова 4096 бітів

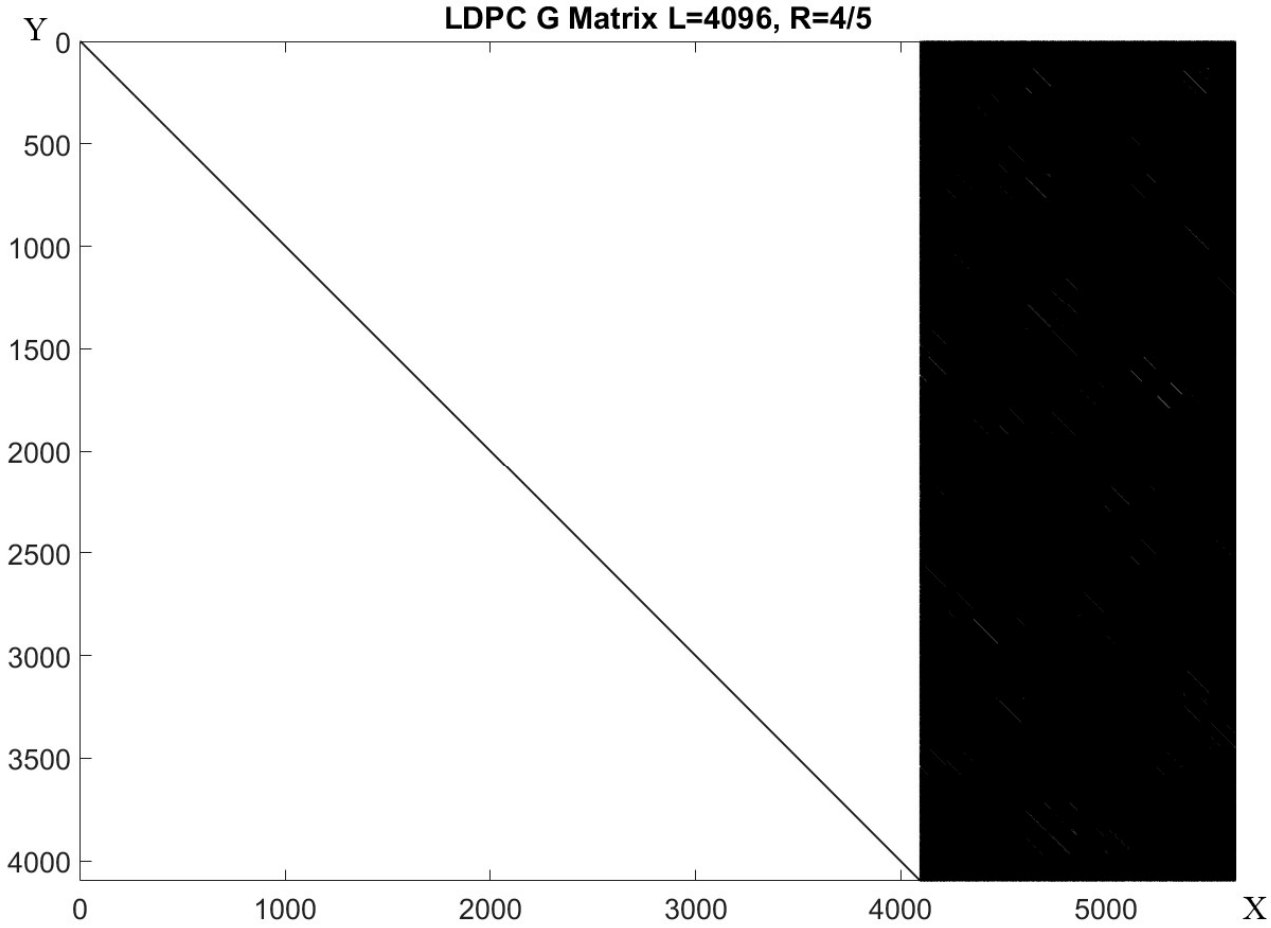

Рисунок 2.7 – Зображення генераторної матриці G LDPC коду для швидкості 4/5 та довжини інформаційного слова 4096 бітів

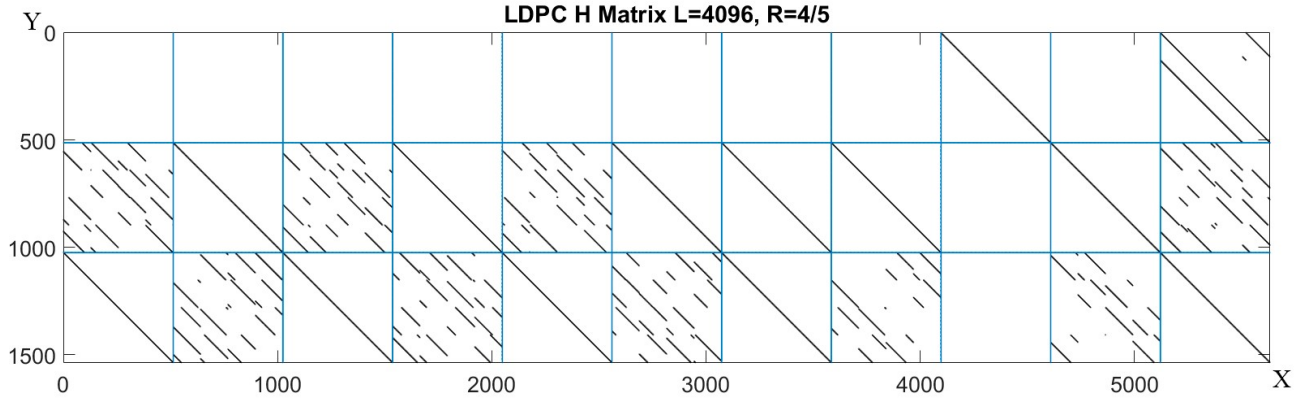

Рисунок 2.8 – Зображення генераторної матриці H LDPC коду для швидкості 4/5 та довжини інформаційного слова 4096 бітів

Можна помітити, що всі матриці мають більше стовбців, а тому і більше перевірочних бітів ніж потрібно для швидкостей 1/2 та 4/5, справа в тому, що останні біти кодового слова проходять процедуру перфорації та не надходять до каналу. А на стороні приймача, перевірочна матриця, сформована спеціальним чином, відновлює ці біти в процесі декодування та користується отриманими бітами надалі при декодуванні.

Відомо, що процедура кодування блокового коду – це лише добуток генераторної матриці з інформаційною послідовністю, в результаті котрого буде отримано кодове слово для передачі по каналу.

Для імплементації у пристрої пропонується використовувати схему кодування, показану на рисунку 2.9.

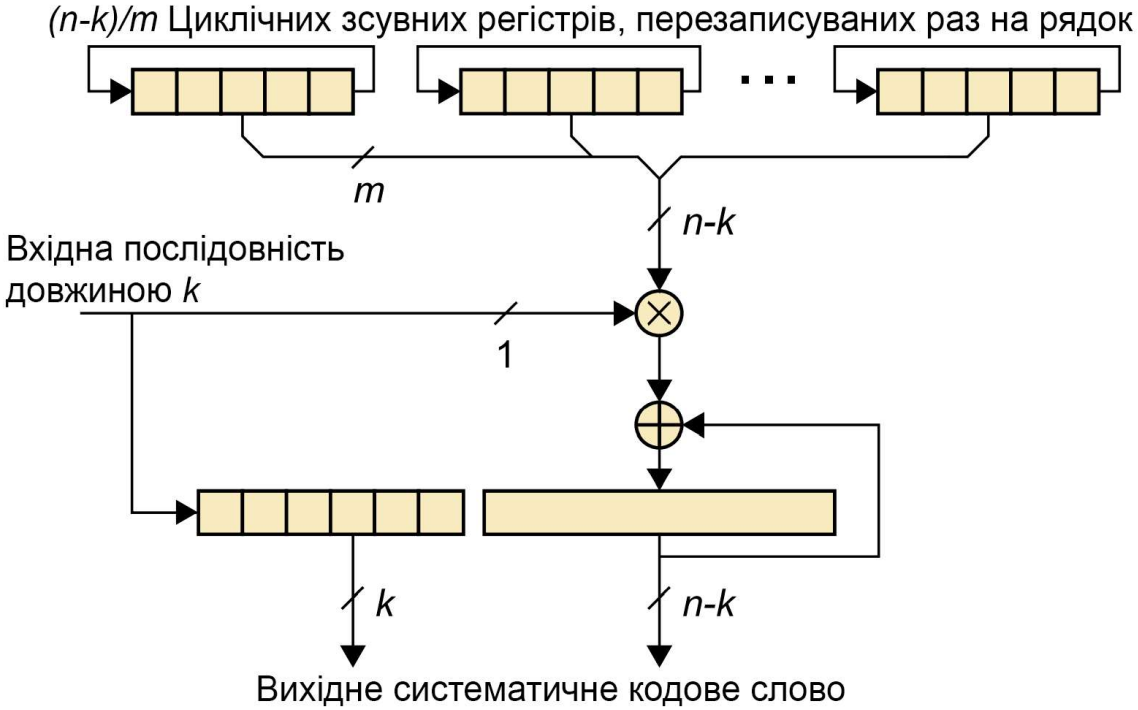

Рисунок 2.9 – LDPC кодер для блокового циркулянтного коду

Для реалізації такої схеми кодування інформаційної послідовності потрібно виділити деяку кількість циклічних зсувних регістрів, які будуть перезаписуватись зсунутими копіями циркулянтних патернів раз на рядок генераторної матриці, та оновлювати ці самі циркулянтні патерни на інші із генераторної матриці в той момент, коли патерн було циклічно зсунуто повністю.

В той момент, коли в інформаційній послідовності зустрічається одиниця – до результуючого вектору перевірочної частини блокового коду за модулем 2 додається послідовність записана на поточний момент в зсувних регістрах.

На виході кодера до інформаційної послідовності приклеюється перевірочна послідовність, отримана сумою за модулем 2 тих рядків перевірочної частини де зустрічалась одиниця в інформаційній частині.

Отриману кодову послідовність було подано до каналу з адитивним білим Гаусовим шумом різної інтенсивності, в результаті чого в кодовій послідовності сталися помилки, котрі намагався виправити LDPC декодер з алгоритмом min-sum. На рисунку 2.10 – рисунку 2.17 показані графіки залежності імовірності бітової помилки від відношення сигнал/шум у порівнянні з послідовністю без завадостійкого кодування.

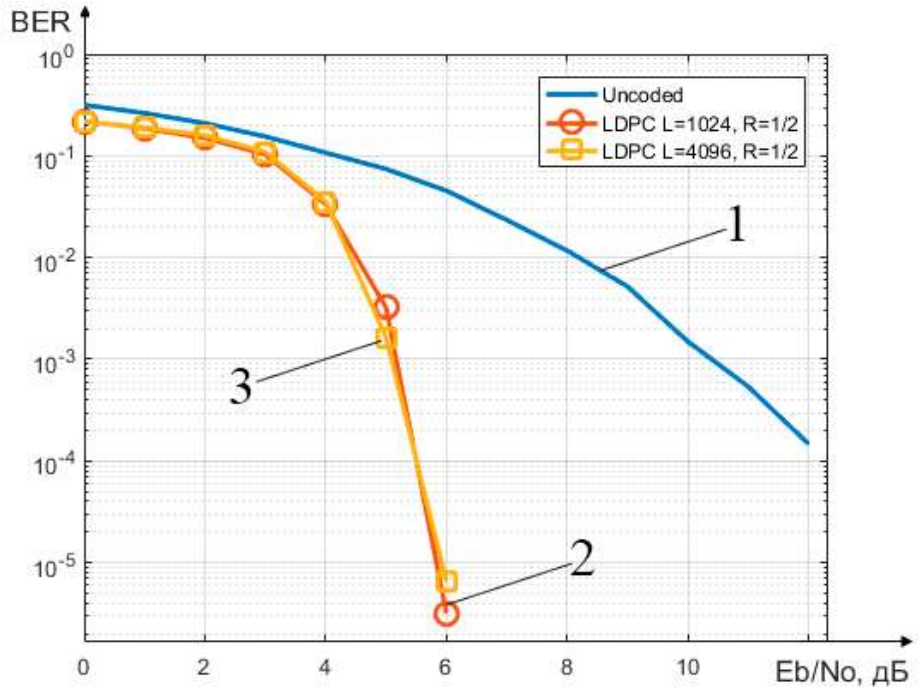

Рисунок 2.10 – Графік залежності бітової помилки від відношення сигнал/шум у випадку різних довжин LDPC коду та швидкості $1/2$:
1 – некодоване;
2 – LDPC код для $L = 1024$ та $R = 1/2$;
3 – LDPC код для $L = 4096$ та $R = 1/2$;

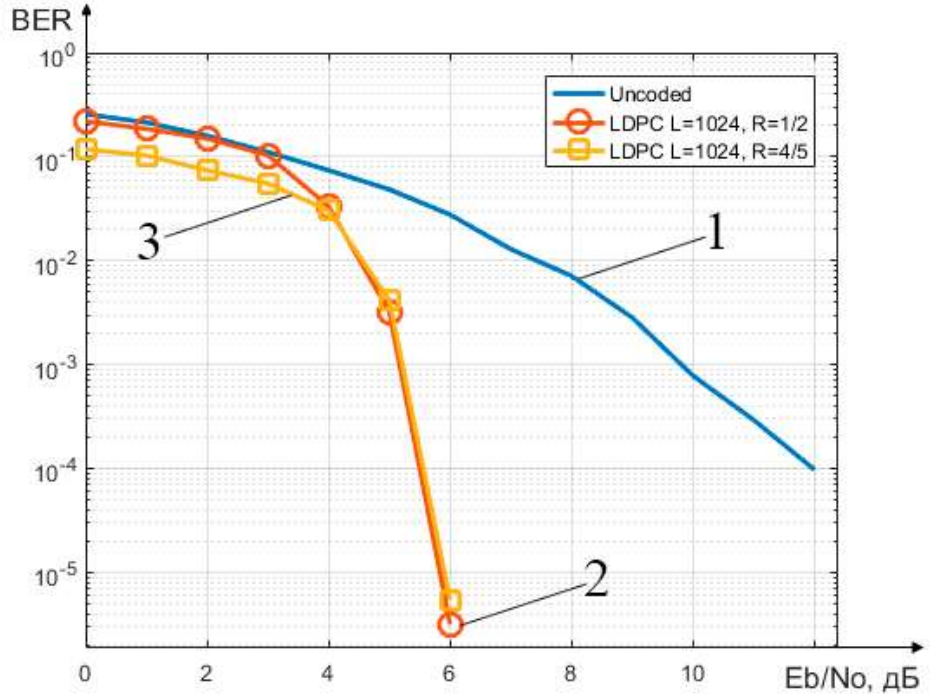

Рисунок 2.11 – Графік залежності бітової помилки від відношення сигнал/шум при довжині LDPC коду 1024 біти та різних швидкостях:
1 – некодоване;
2 – LDPC код для $L = 1024$ та $R = 1/2$;
3 – LDPC код для $L = 1024$ та $R = 4/5$;

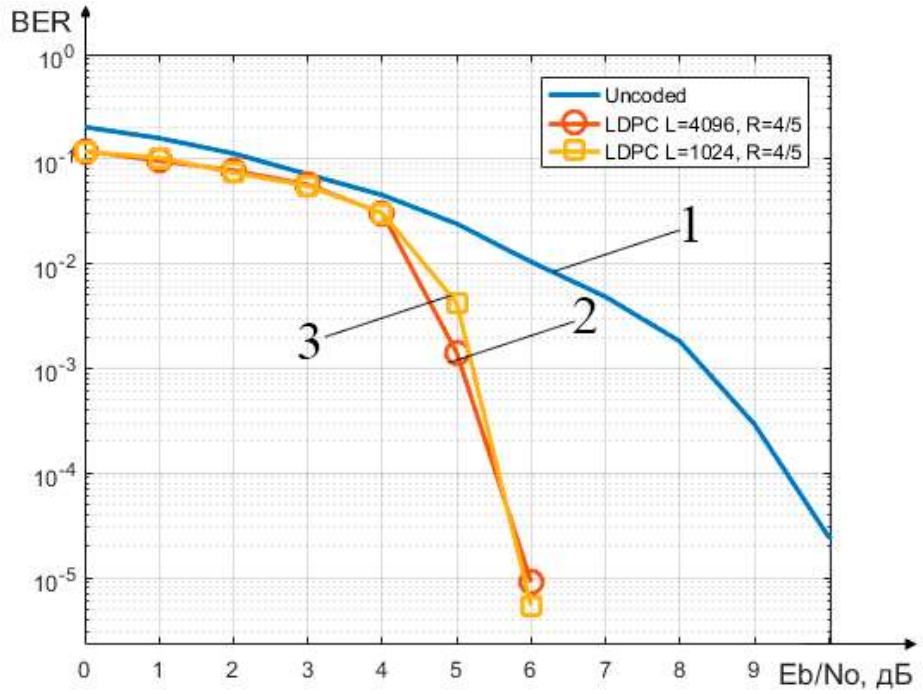

Рисунок 2.12 – Графік залежності бітової помилки від відношення сигнал/шум у випадку різних довжин LDPC коду та швидкості $4/5$:
1 – некодоване;
2 – LDPC код для $L = 4096$ та $R = 4/5$;
3 – LDPC код для $L = 1024$ та $R = 4/5$;

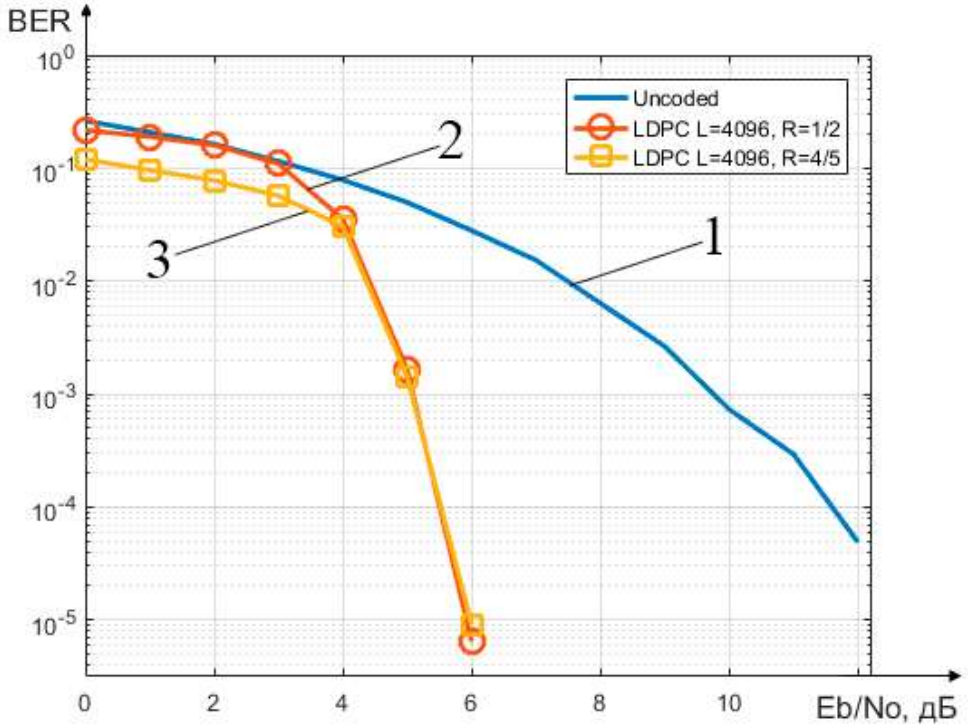

Рисунок 2.13 – Графік залежності бітової помилки від відношення сигнал/шум при довжині LDPC коду 4096 бітів та різних швидкостях:
1 – некодоване;
2 – LDPC код для $L = 4096$ та $R = 1/2$;
3 – LDPC код для $L = 4096$ та $R = 4/5$;

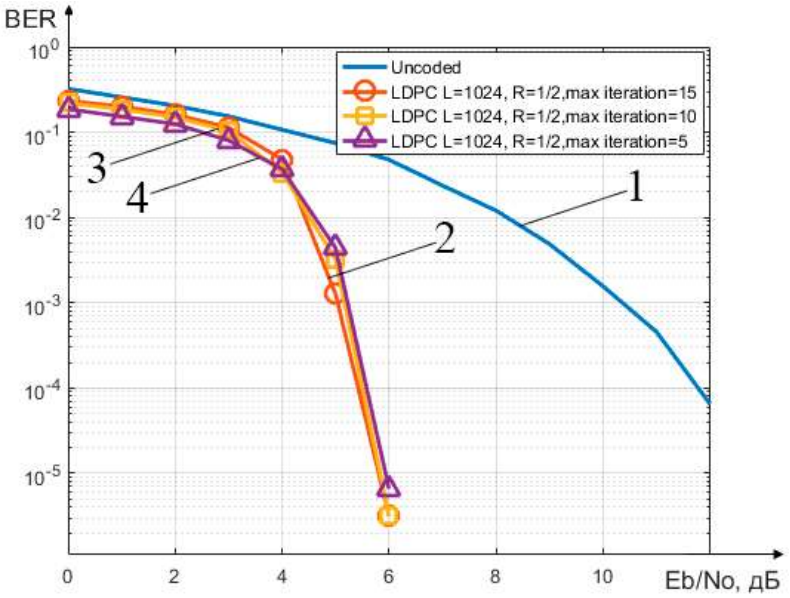

Рисунок 2.14 – Графік залежності бітової помилки від відношення сигнал/шум при довжині LDPC коду 1024 біти, швидкості $1/2$ та різній кількості ітерацій:

1 – некодоване;

2 – LDPC код для $L = 1024$, $R = 1/2$, кількість ітерацій 15;

3 – LDPC код для $L = 1024$, $R = 1/2$, кількість ітерацій 10;

4 – LDPC код для $L = 1024$, $R = 1/2$, кількість ітерацій 5

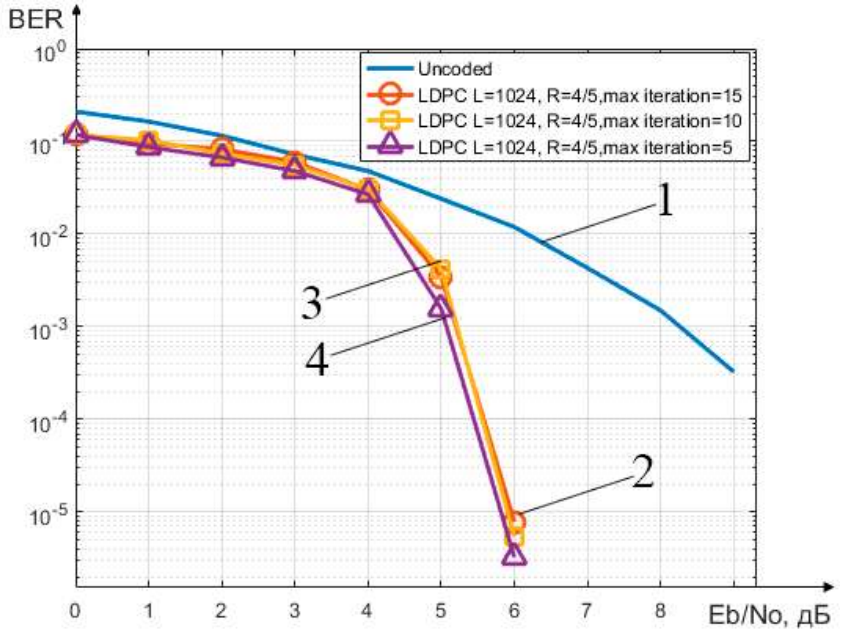

Рисунок 2.15 – Графік залежності бітової помилки від відношення сигнал/шум при довжині LDPC коду 1024 біти, швидкості $4/5$ та різній кількості ітерацій:

1 – некодоване;

2 – LDPC код для $L = 1024$, $R = 4/5$, кількість ітерацій 15;

3 – LDPC код для $L = 1024$, $R = 4/5$, кількість ітерацій 10;

4 – LDPC код для $L = 1024$, $R = 4/5$, кількість ітерацій 5

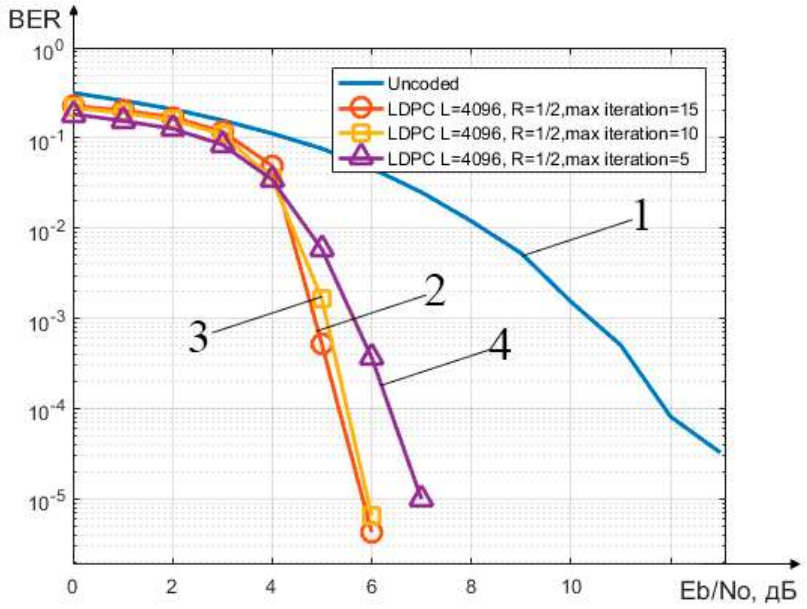

Рисунок 2.16 – Графік залежності бітової помилки від відношення сигнал/шум при довжині LDPC коду 4096 біти, швидкості $1/2$ та різній кількості ітерацій:

1 – некодоване;

2 – LDPC код для $L = 4096$, $R = 1/2$, кількість ітерацій 15;

3 – LDPC код для $L = 4096$, $R = 1/2$, кількість ітерацій 10;

4 – LDPC код для $L = 4096$, $R = 1/2$, кількість ітерацій 5

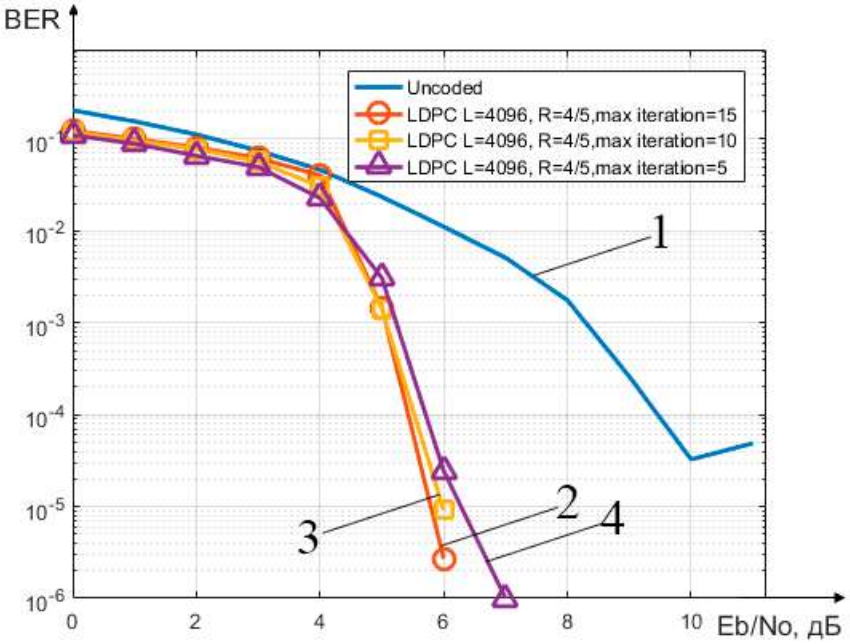

Рисунок 2.17 – Графік залежності бітової помилки від відношення сигнал/шум при довжині LDPC коду 4096 біти, швидкості $4/5$ та різній кількості ітерацій:

1 – некодоване;

2 – LDPC код для $L = 4096$, $R = 4/5$, кількість ітерацій 15;

3 – LDPC код для $L = 4096$, $R = 4/5$, кількість ітерацій 10;

4 – LDPC код для $L = 4096$, $R = 4/5$, кількість ітерацій 5

Із графіків залежності імовірності бітової помилки від відношення сигнал/шум можна зробити декілька висновків.

По-перше, LDPC кодування із застосуванням кодів із сімейства AR4JA дозволяє зменшити імовірність бітової помилки навіть в поганих каналах, де відношення сигнал/шум дуже низьке і є велика імовірність помилки в каналі. Деякі завадостійкі коди в процесі декодування сильно змінених кодових слів, отриманих з поганого каналу, можуть зробити ще більше помилок в кодовій послідовності під час декодування. Але обрані LDPC коди гарантують меншу імовірність бітової помилки при відношенні сигнал/шум хоча б більше нуля.

По-друге, при застосуванні LDPC кодів з сімейства AR4JA гарантується енергетичний виграш щонайменше 3.2 дБ (випадок коду довжини 1024, швидкості $4/5$ та максимальній кількості ітерацій 5) при імовірності бітової помилки 10^{-3}. Більшість ліній, які відображають залежність імовірності бітової помилки від відношення сигнал шум після декодування не мають значень більше 5 дБ на графіках тому, що імовірність бітової помилки в таких випадках дорівнює майже нулю при великій кількості переданих через канал кодових слів, тобто у великій кількості випадків LDPC кодом було виправлено всі помилки, які траплялися на достатньо високих для цього відношеннях сигнал/шум.

По-третє, слід відмітити, що різна довжина коду майже ніяким чином не впливає на імовірність бітової помилки в умовах поганих каналів, але із збільшенням відношення сигнал/шум стає помітна менша імовірність бітової помилки при застосування кодів більшої довжини на однаковому рівні відношення сигнал/шум. Це помітно для двох швидкостей $1/2$ і $4/5$ на рисунках 2.10 і 2.12, де різниця імовірностей бітової помилки може складати пів порядку при відношенні сигнал/шум 5 дБ.

По-четверте, менша імовірність бітової помилки при використанні кодів з більшою швидкістю, тобто $4/5$, порівняно з кодом такої ж довжини, але меншої швидкості, в поганих каналах (рисунок 2.11), (рисунок 2.13), може пояснюватися двома версіями. За першою версією, в кодах з більшою швидкістю просто менше перевірочних бітів, в яких також можуть статися помилки і які надалі будуть впливати на декодування. За другою версією, в коді з більшою швидкістю менше перевірок на парність, але вони містять більшу кількість бітів, що в них входять, тому процес декодування робить нові помилки в кодовому слові, в спробах виправити послідовність з поганого каналу, з меншою ефективністю, ніж в коді з більшою кількістю перевірок на парність, тобто в коді зі швидкістю $1/2$.

У п’ятих, різна кількість ітерацій також впливає на енергетичний виграш, який дає LDPC кодування, але для кожної довжини та швидкості є свої нюанси. Так, в результаті моделювання стало ясно, що для швидкості коду $1/2$ (будь-якої довжини інформаційного блоку) збільшення максимальної кількості ітерацій – зменшує імовірність бітової помилки, і чим більша довжина коду, тим більше це помітно. Наприклад, на рисунку 2.16, при відношенні сигнал/шум 5 дБ, різниця між декодуванням з кількістю ітерацій 5 та 15 становить цілий порядок.

Таким чином, застосування LDPC кодів із сімейства AR4JA дозволяє знизити потужність передавача або зменшити час на передачу одного символу,

щонайменше у 2 рази та отримати ту ж імовірність бітової помилки, що була б отримана без застосування завадостійкого LDPC кодування. І такий енергетичний виграш можна отримати лише у випадку довжини коду 1024 біти та швидкості $4/5$, а чим більша довжина коду та менша його швидкість, тим більший можна отримати енергетичний виграш.

2.2 Математичне моделювання турбо-кодування

Як вже було сказано, турбо-код – це комбінація двох простих рекурсивних згорткових кодів, кожен з яких використовує невелику кількість станів. Ці прості згорткові коди насправді є «закінченими» згортковими кодами і, отже, блоковими кодами. Для блоку k інформаційних бітів кожен складовий код генерує набір бітів парності. Турбо-код складається з інформаційних бітів та обох наборів парності.

Ключовим нововведенням є перемежувач P, який перетворює k інформаційних бітів перед кодуванням другого коду. Якщо перемежувач підібрано правильно, то інформаційні блоки, які відповідають кодовим словам, схильним до помилок, в одному коді відповідатимуть кодовим словам, стійким до помилок, в іншому коді. Отриманий код досягає продуктивності, подібної до загальновідомих «випадкових» кодів Шеннона, але випадкові коди наближаються до оптимальної продуктивності лише ціною надмірно складного декодера. Конкретний турбокодер у рекомендованому стандарті CCSDS [51] детально показаний на рисунку 2.18.

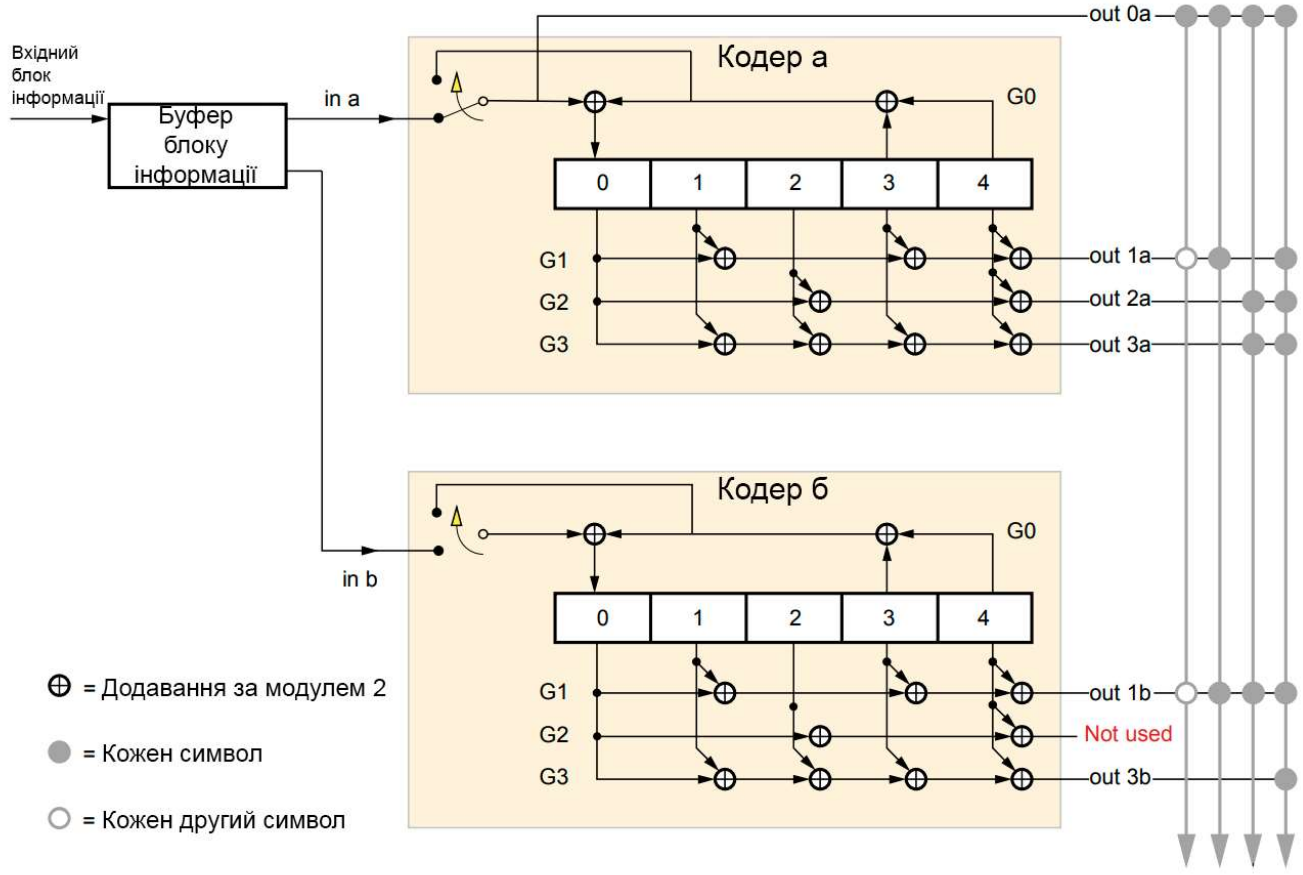

Рисунок 2.18 – Структурна схема рекомендованого CCSDS турбо-кодера

Тут два згорткових кодери в рекомендованому стандарті [51] є рекурсивними з довжиною обмеження $K = 5$ і реалізуються за допомогою регістрів зсуву зворотного зв'язку. Однак, на відміну від звичайного згорткового коду, кодове слово турбо-коду завершується запуском кожного кодера протягом додаткових $K - 1$ разів після кінця кадру інформаційного біта. Після кодування останнього біта в кадрі крайній лівий суматор у кожному компонентному кодері отримує дві копії того самого біта зворотного зв'язку, що призводить до нульового виходу. Після $K - 1$ разів усі 4 комірки пам'яті заповнюються нулями, але в цей момент кодер ще продовжує видавати ненульові закодовані символи.

Рекомендований код у [51] підтримує швидкості $1/3$, $1/4$ та $1/6$, без використання перфорації бітів. При використанні перфорації бітів можна досягти швидкості $1/2$. Перемежувач, який рекомендовано в [51] побудовано спеціальним принципом і позиції перестановок може бути обчислено у процесі кодування або обчислено завчасно та збережено для подальшого кодування. Пережувач підтримує інформаційні блоки довжиною від 1784 бітів до 16384 бітів, але в математичному моделюванні було використано лише коди довжиною 1784, 3568, 7136 та 8920 бітів з використанням двох швидкостей: $1/2$ за рахунок перфорації бітів, та $1/3$.

Кожен компонентний кодер має гратчасту діаграму показану на рисунку 2.19. Діаграма містить в собі 16 станів та має $k + 4$ довжину.

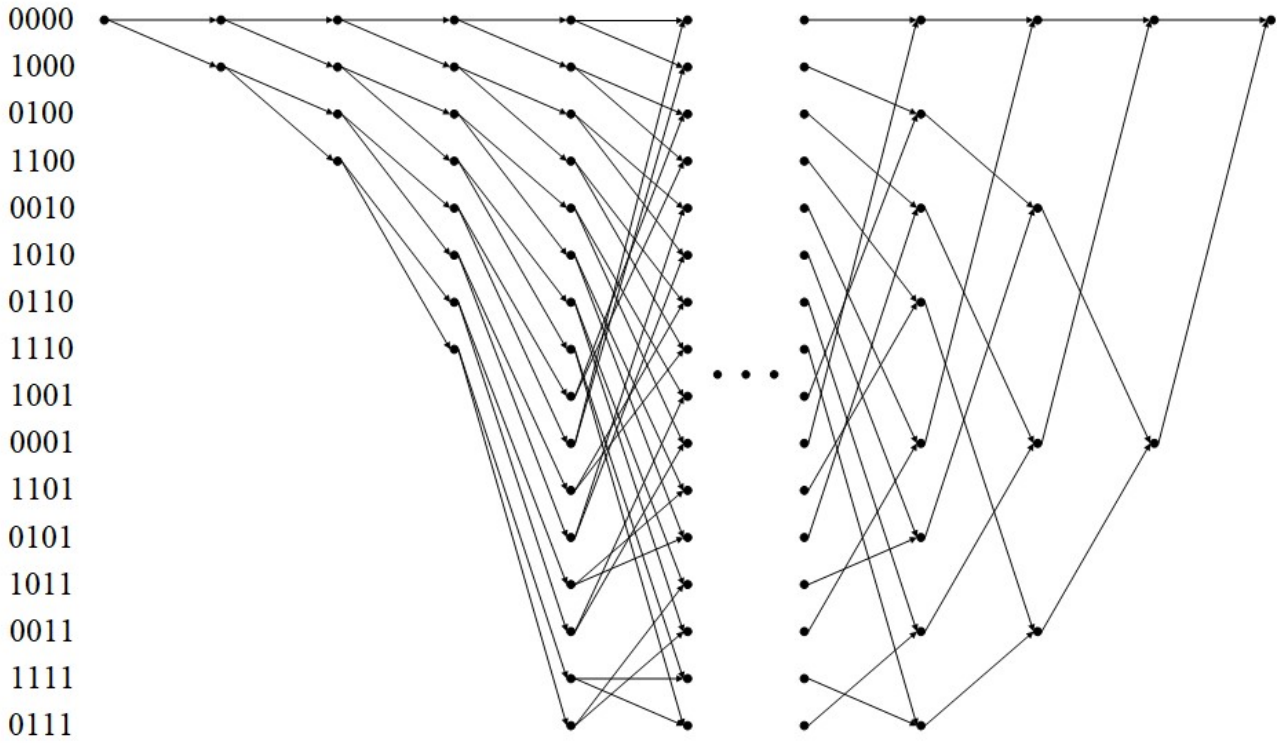

Рисунок 2.19 – Гратчаста діаграма рекомендованого компонентного згорткового коду

Структурну схему декодера показано на рисунку 2.20. Сам декодер складається з двох компонентних згорткових декодерів, а в якості алгоритму декодування з м'яким виходом у моделюванні використовувався алгоритм BCJR.

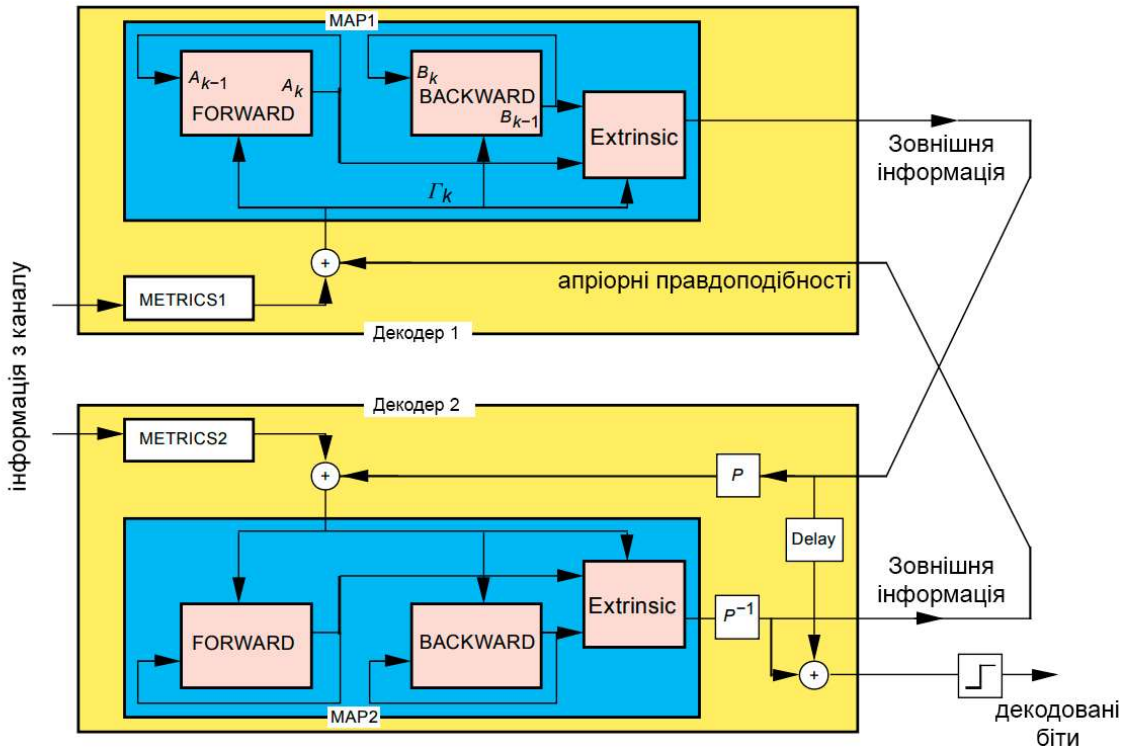

Рисунок 2.20 – Структурна схема рекомендованого турбо декодера

Отриману в результаті кодування турбо-кодером послідовність було подано до каналу з адитивним білим Гаусовим шумом різної інтенсивності, в результаті чого в кодовій послідовності сталися помилки, які вирішувалися турбо-декодером. На рисунку 2.21 – рисунку 2.34 показані графіки залежності імовірності бітової помилки від відношення сигнал/шум у порівнянні з послідовністю без завадостійкого кодування.

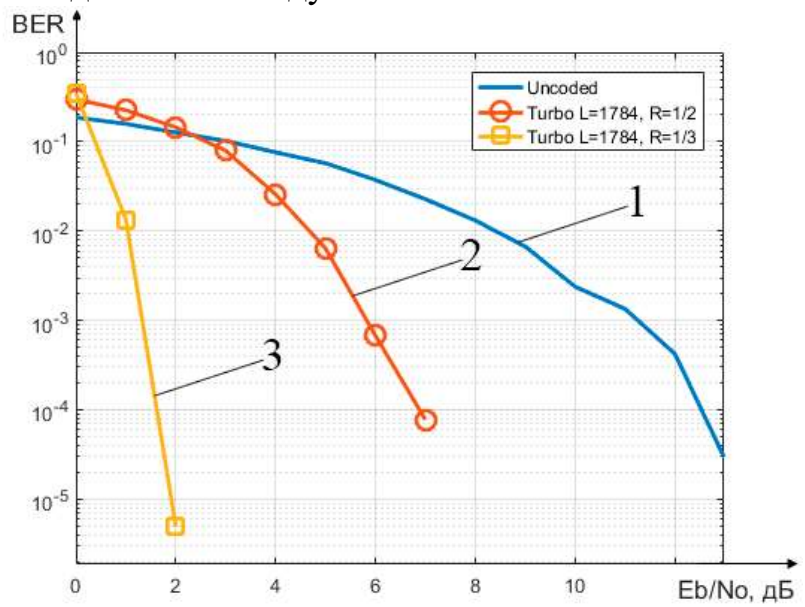

Рисунок 2.21 – Графік залежності імовірності бітової помилки турбо-коду від відношення сигнал/шум при довжині блоку 1784 і різних швидкостях:
1 – некодоване;
2 – турбо код для $L = 1784$ та $R = 1/2$;
3 – турбо код для $L = 1784$ та $R = 1/3$

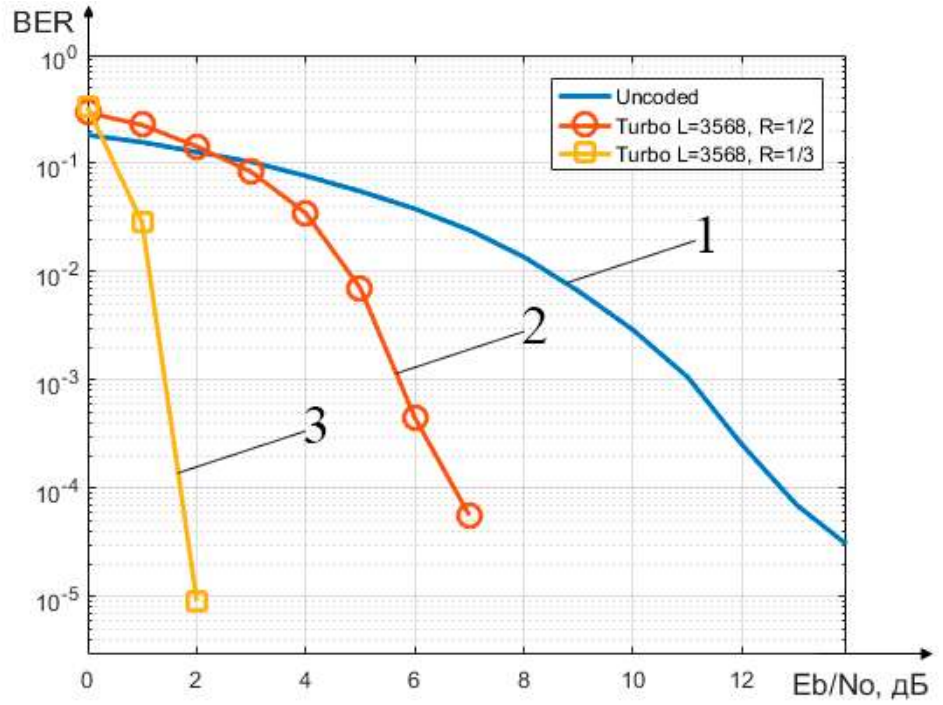

Рисунок 2.22 – Графік залежності імовірності бітової помилки турбо-коду від відношення сигнал/шум при довжині блоку 3568 і різних швидкостях:
1 – некодоване;
2 – турбо код для $L = 3568$ та $R = 1/2$;
3 – турбо код для $L = 3568$ та $R = 1/3$

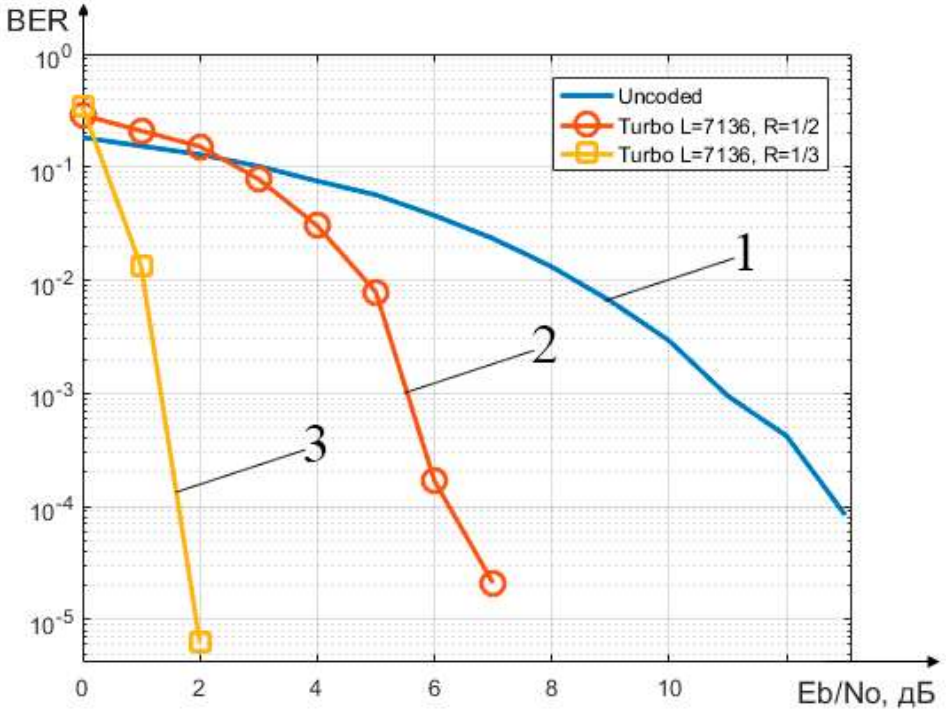

Рисунок 2.23 – Графік залежності імовірності бітової помилки турбо-коду від відношення сигнал/шум при довжині блоку 7136 і різних швидкостях:
1 – некодоване;
2 – турбо код для $L = 7136$ та $R = 1/2$;
3 – турбо код для $L = 7136$ та $R = 1/3$

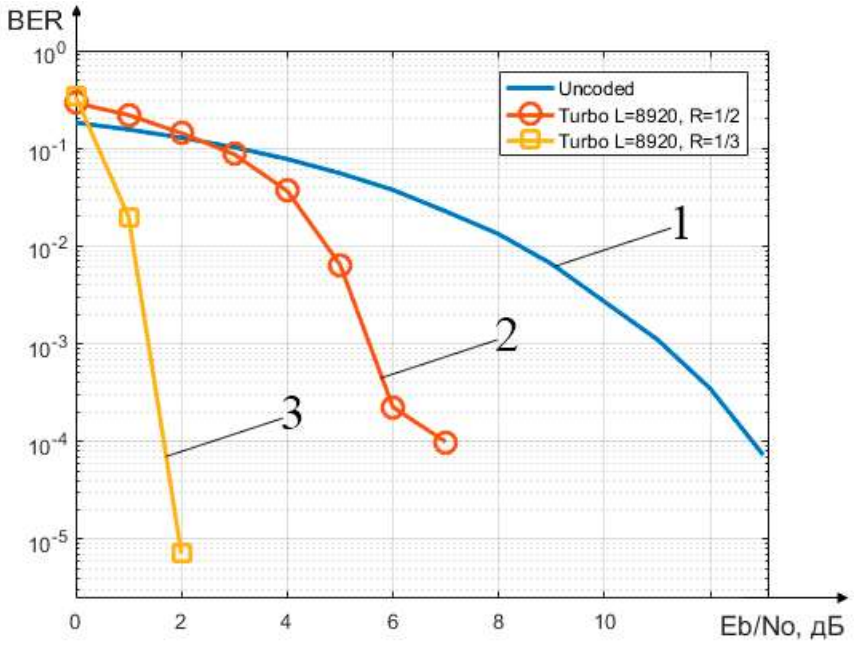

Рисунок 2.24 – Графік залежності імовірності бітової помилки турбо-коду від відношення сигнал/шум при довжині блоку 8920 і різних швидкостях:
1 – некодоване;
2 – турбо код для $L = 8920$ та $R = 1/2$;
3 – турбо код для $L = 8920$ та $R = 1/3$

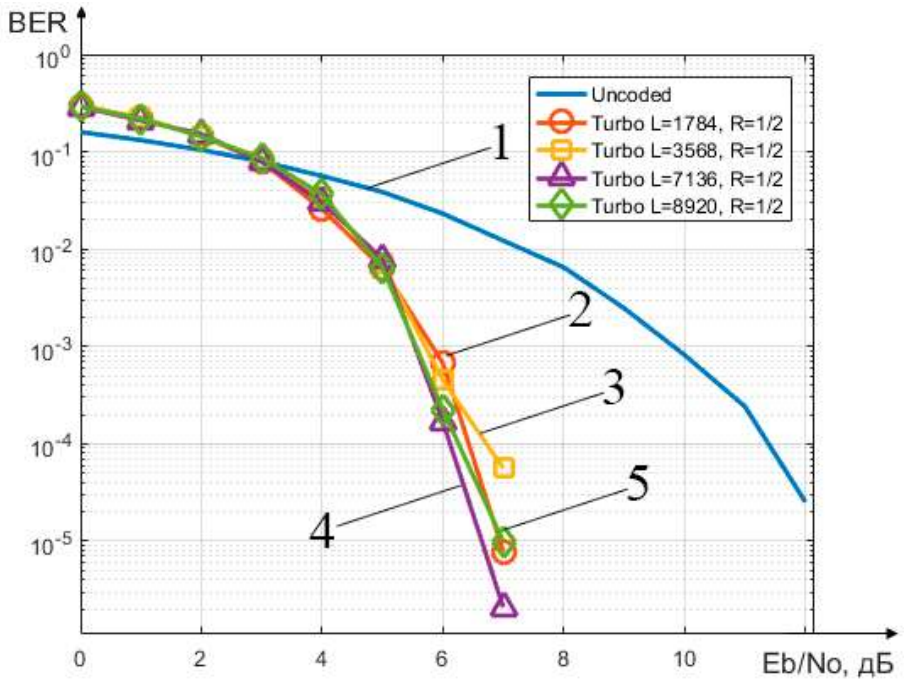

Рисунок 2.25 – Графік залежності імовірності бітової помилки турбо-коду від відношення сигнал/шум при швидкості $1/2$ і різній довжині блоку:
1 – некодоване;
2 – турбо код для $L = 1784$ та $R = 1/2$;
3 – турбо код для $L = 3568$ та $R = 1/2$;
4 – турбо код для $L = 7136$ та $R = 1/2$;
5 – турбо код для $L = 8920$ та $R = 1/2$

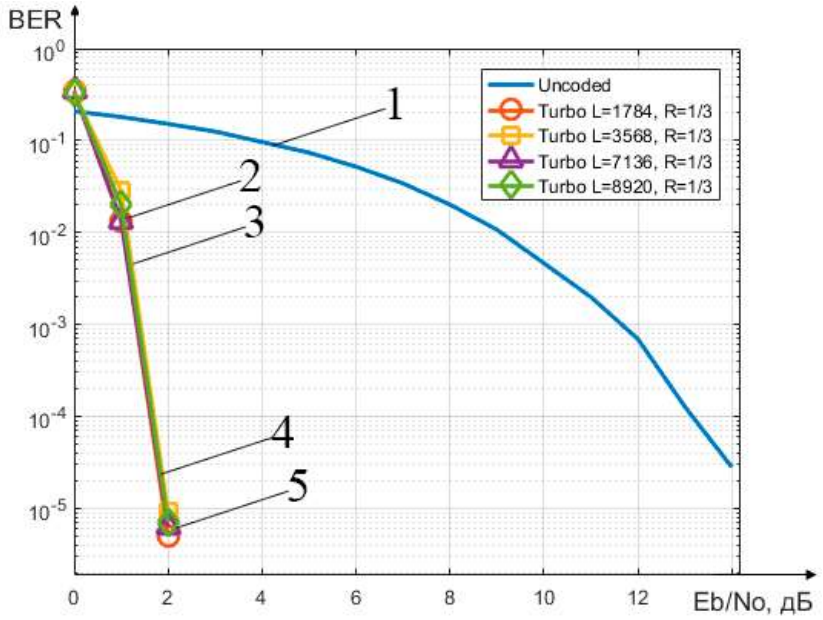

Рисунок 2.26 – Графік залежності імовірності бітової помилки турбо-коду від відношення сигнал/шум при швидкості $1/3$ і різній довжині блоку:
1 – некодоване;
2 – турбо код для $L = 1784$ та $R = 1/3$;
3 – турбо код для $L = 3568$ та $R = 1/3$;
4 – турбо код для $L = 7136$ та $R = 1/3$;
5 – турбо код для $L = 8920$ та $R = 1/3$

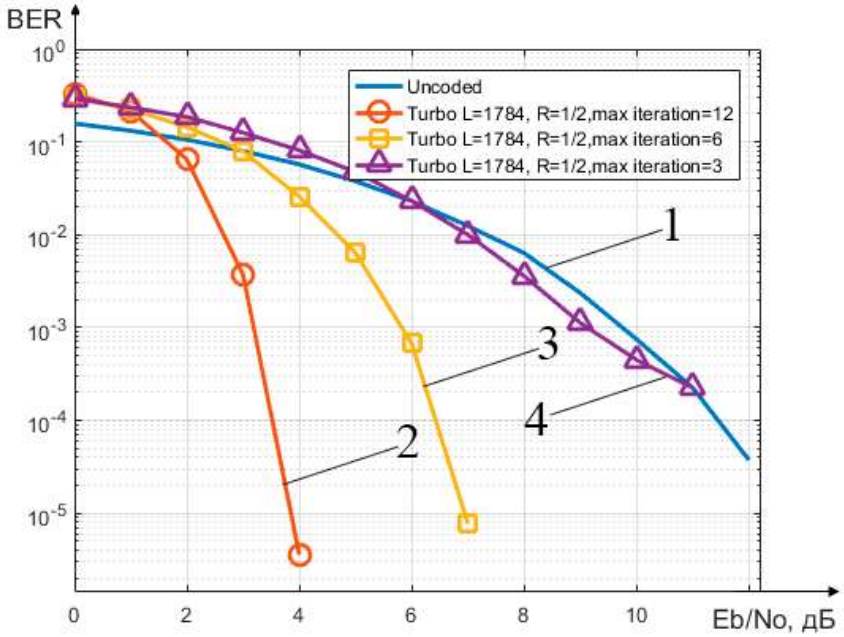

Рисунок 2.27 – Графік залежності імовірності бітової помилки турбо-коду від відношення сигнал/шум при довжині блоку 1784, швидкості $1/2$ та різній кількості ітерацій:
1 – некодоване;
2 – турбо код для $L = 1784$, $R = 1/2$, кількість ітерацій 12;
3 – турбо код для $L = 1784$, $R = 1/2$, кількість ітерацій 6;
4 – турбо код для $L = 1784$, $R = 1/2$, кількість ітерацій 3

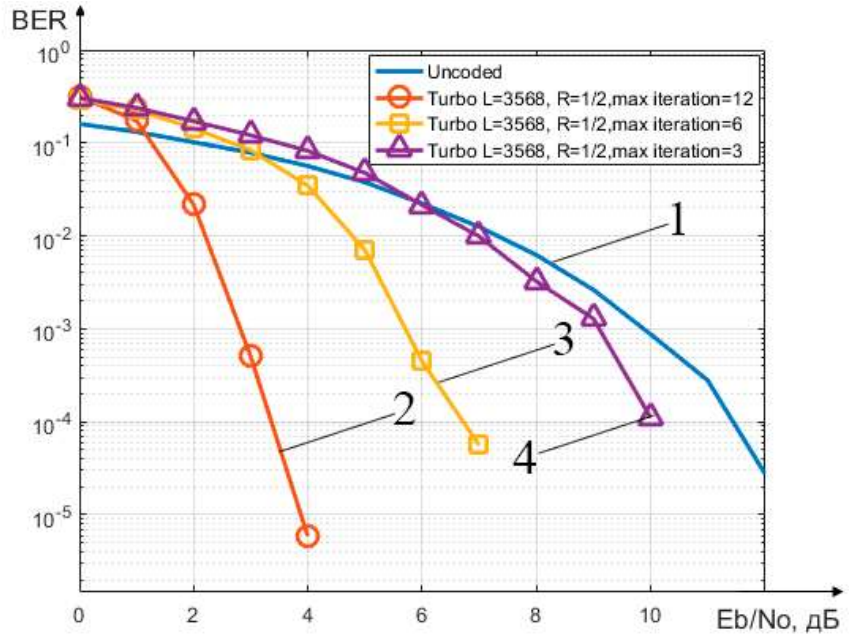

Рисунок 2.28 – Графік залежності імовірності бітової помилки турбо-коду від відношення сигнал/шум при довжині блоку 3568, швидкості $1/2$ та різній кількості ітерацій:

1 – некодоване;

2 – турбо код для $L = 3568$, $R = 1/2$, кількість ітерацій 12;

3 – турбо код для $L = 3568$, $R = 1/2$, кількість ітерацій 6;

4 – турбо код для $L = 3568$, $R = 1/2$, кількість ітерацій 3

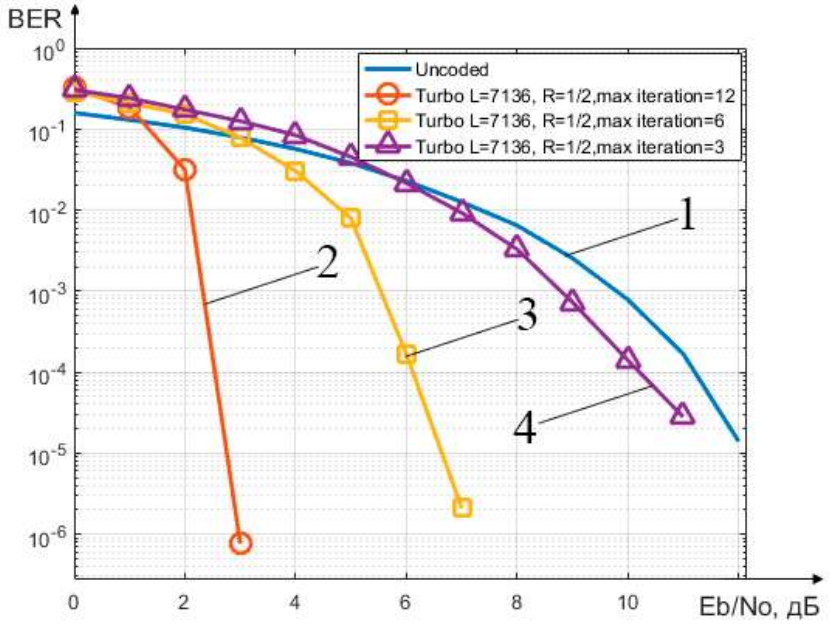

Рисунок 2.29 – Графік залежності імовірності бітової помилки турбо-коду від відношення сигнал/шум при довжині блоку 7136, швидкості $1/2$ та різній кількості ітерацій:

1 – некодоване;

2 – турбо код для $L = 7136$, $R = 1/2$, кількість ітерацій 12;

3 – турбо код для $L = 7136$, $R = 1/2$, кількість ітерацій 6;

4 – турбо код для $L = 7136$, $R = 1/2$, кількість ітерацій 3

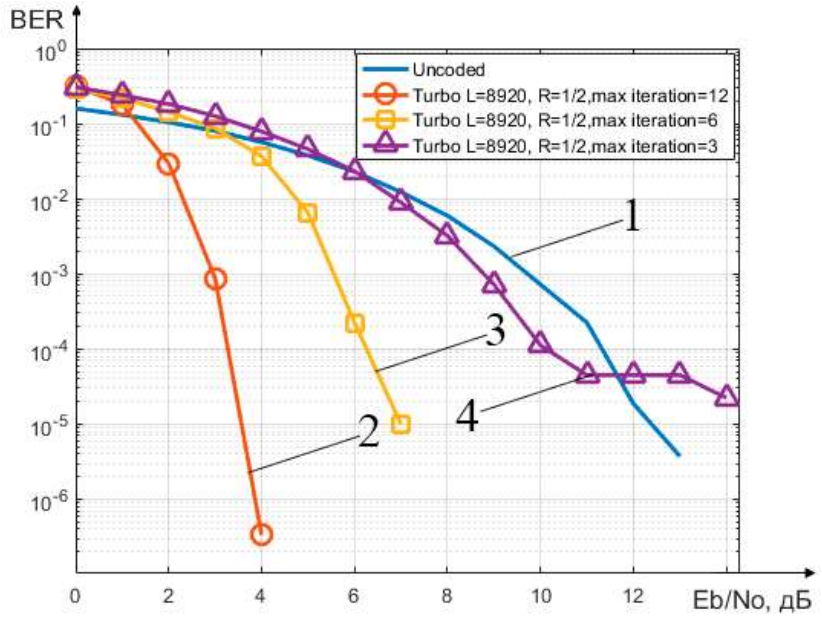

Рисунок 2.30 – Графік залежності імовірності бітової помилки турбо-коду від відношення сигнал/шум при довжині блоку 8920, швидкості $1/2$ та різній кількості ітерацій:

1 – некодоване;

2 – турбо код для $L = 8920$, $R = 1/2$, кількість ітерацій 12;

3 – турбо код для $L = 8920$, $R = 1/2$, кількість ітерацій 6;

4 – турбо код для $L = 8920$, $R = 1/2$, кількість ітерацій 3

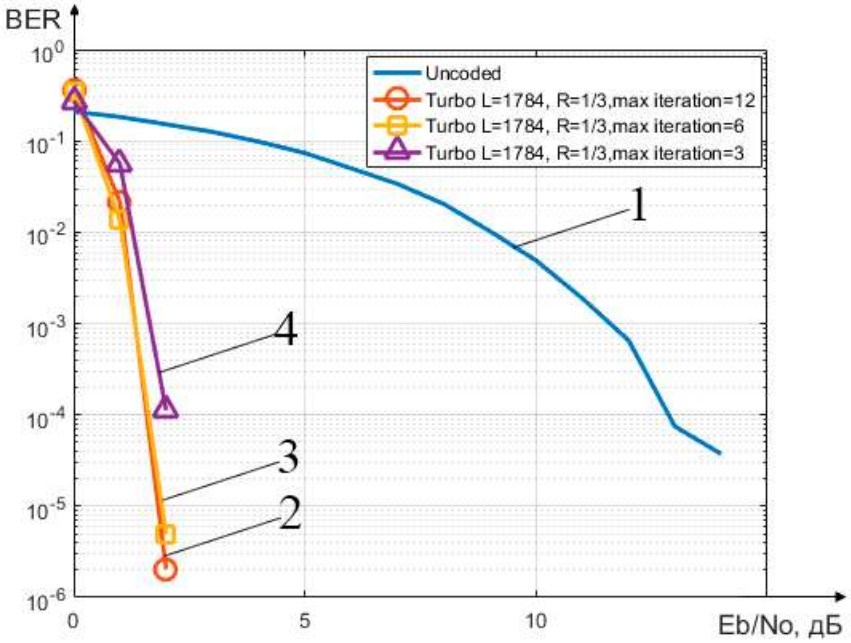

Рисунок 2.31 – Графік залежності імовірності бітової помилки турбо-коду від відношення сигнал/шум при довжині блоку 1784, швидкості $1/3$ та різній кількості ітерацій:

1 – некодоване;

2 – турбо код для $L = 1784$, $R = 1/3$, кількість ітерацій 12;

3 – турбо код для $L = 1784$, $R = 1/3$, кількість ітерацій 6;

4 – турбо код для $L = 1784$, $R = 1/3$, кількість ітерацій 3

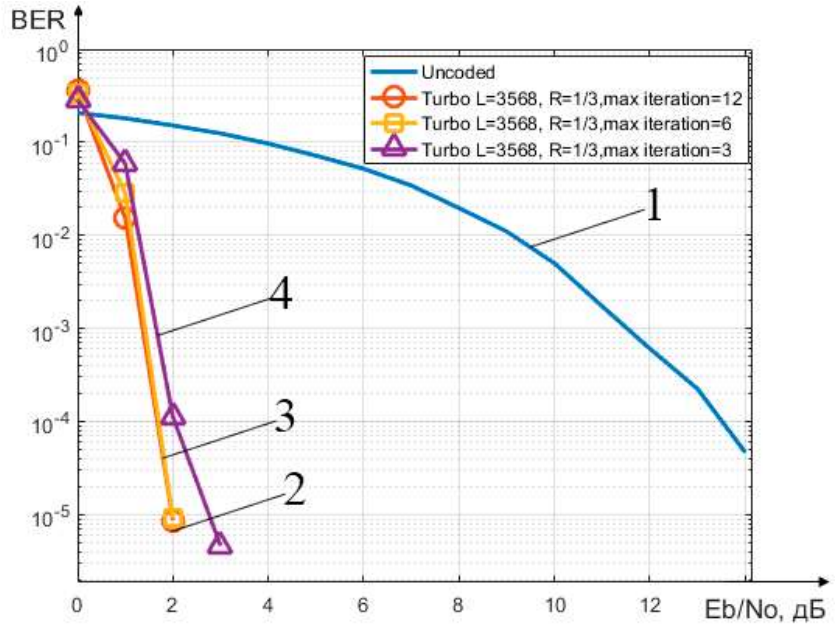

Рисунок 2.32 – Графік залежності імовірності бітової помилки турбо-коду від відношення сигнал/шум при довжині блоку 3568, швидкості $1/3$ та різній кількості ітерацій:

1 – некодоване;

2 – турбо код для $L = 3568$, $R = 1/3$, кількість ітерацій 12;

3 – турбо код для $L = 3568$, $R = 1/3$, кількість ітерацій 6;

4 – турбо код для $L = 3568$, $R = 1/3$, кількість ітерацій 3

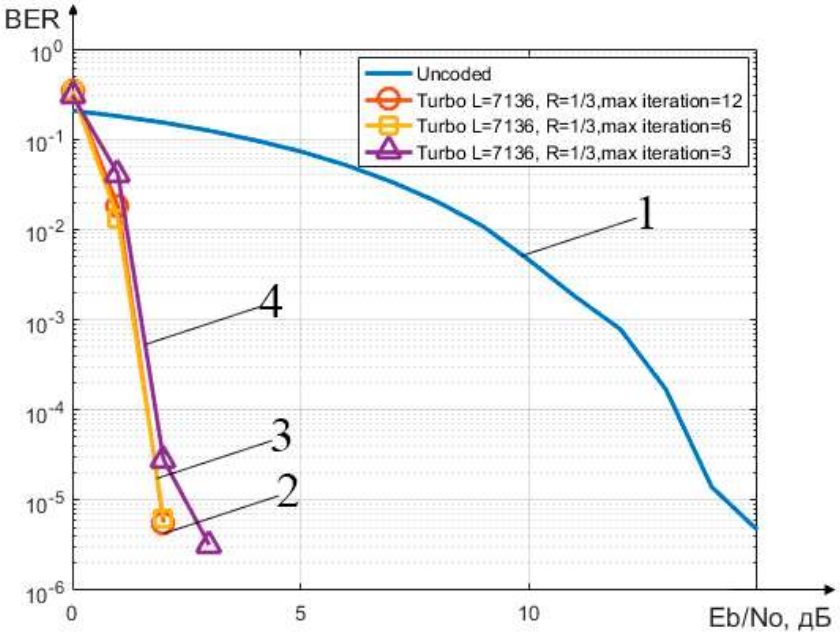

Рисунок 2.33 – Графік залежності імовірності бітової помилки турбо-коду від відношення сигнал/шум при довжині блоку 7136, швидкості $1/3$ та різній кількості ітерацій:

1 – некодоване;

2 – турбо код для $L = 7136$, $R = 1/3$, кількість ітерацій 12;

3 – турбо код для $L = 7136$, $R = 1/3$, кількість ітерацій 6;

4 – турбо код для $L = 7136$, $R = 1/3$, кількість ітерацій 3

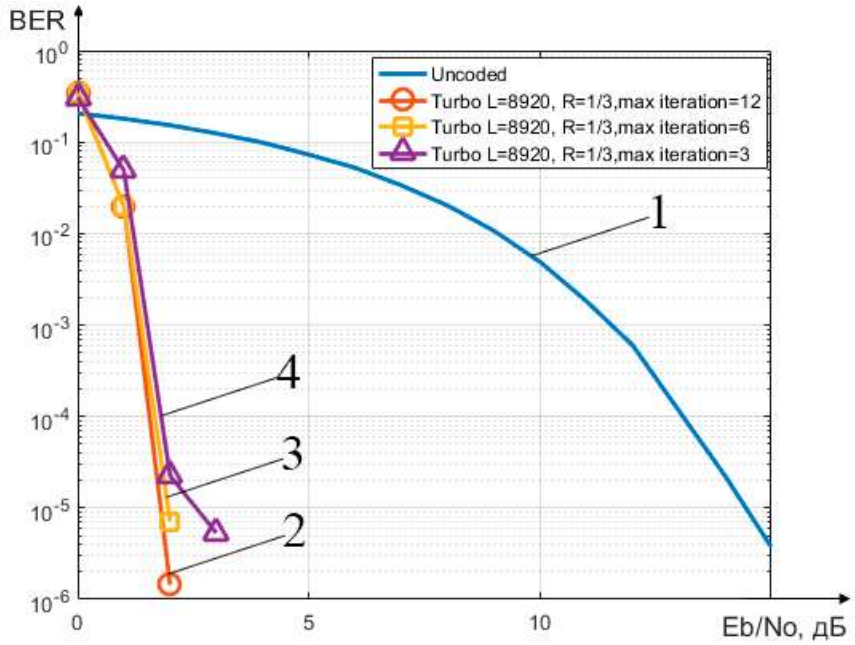

Рисунок 2.34 – Графік залежності імовірності бітової помилки турбо-коду від відношення сигнал/шум при довжині блоку 8920, швидкості $1/3$ та різній кількості ітерацій:

1 – некодоване;

2 – турбо код для $L = 8920$, $R = 1/3$, кількість ітерацій 12;

3 – турбо код для $L = 8920$, $R = 1/3$, кількість ітерацій 6;

4 – турбо код для $L = 8920$, $R = 1/3$, кількість ітерацій 3

З отриманих графіків залежності імовірності бітової помилки від відношення сигнал/шум шляхом математичного моделювання можна виділити наступні властивості турбо-кодів.

Порівняння коду однієї довжини та різних швидкостей ($1/3$ і $1/2$) показує велику залежність коду від цього параметру. Швидкість $1/2$ досягається за рахунок перфорації половини перевірочних бітів з кодового слова швидкості $1/3$, а на стороні приймача виколоті біти заповнюються нульовими значеннями. Таке прискорення коду має сильний вплив на його корегуючі властивості в сторону їх погіршення.

Енергетичний виграш, який дає застосування турбо-коду у найгіршому випадку, тобто при використанні інформаційного блоку довжиною 1784 при швидкості $1/2$ і максимальній кількості повних ітерацій декодера – 3, складає зовсім мізерну величину, але в гарному каналі, де імовірність помилки невелика, турбо-кодування все ж краще застосовувати. У випадку з 6 повними ітераціями декодера картина вже краща і на рівні імовірності помилки в каналі 10^{-3} можна отримати енергетичний виграш приблизно в 3.8 дБ.

У зовсім поганих каналах не слід використовувати турбо-кодування, тому що затримка до отримання декодованої послідовності збільшується, порівняно з послідовністю без кодування, а імовірність помилки після декодування залишається такою ж як до кодування або стає вище під час декодування послідовності з великою кількістю помилок.

Різна максимальна кількість ітерацій декодера грає величезну роль у передачі повідомлень зі швидкістю коду $1/2$, а при використанні кодів зі швидкістю $1/3$ декодер має відмінну ефективність і швидко насичується та декодує повідомлення. Наприклад, на рисунку 2.30 можна помітити «полицю», яку було згадано в першому розділі цієї роботи, при застосуванні турбо-коду із значенням 3 максимальної кількості ітерацій декодування.

При порівнянні кодів з різною довжиною інформаційного блоку можна помітити, що чим більше інформаційний блок, тим нижча імовірність помилки в декодованій послідовності. Це правило можна застосувати до обох швидкостей коду, але при використанні кодів зі швидкістю $1/3$ імовірність бітової помилки наближується до межі Шеннона, тому різницю помітити важко.

Отже, застосування завадостійких турбо-кодів зі швидкістю $1/3$ дозволяє знизити енергетичні затрати на передачу повідомлень приблизно на 10.5 дБ та отримати імовірність бітової помилки на рівні 10^{-4} після декодування, і саме тому ці коди рекомендовані до застосування в радіолініях зв'язку з космічними апаратами.

2.3 Порівняльний аналіз ефективності турбо-кодування та LDPC-кодування

Рекомендовані до використання турбо-коди та LDPC коди не мають однакових параметрів, окрім швидкості $1/2$. Різні довжини у різних кодів очевидно дають різну ефективність та результати. Тому в деяких випадках порівняльного аналізу було вирішено максимально наскільки це можливо наблизити один код до іншого. Таким чином, результати математичного моделювання та порівняння показані на рисунках 2.35 – 2.39.

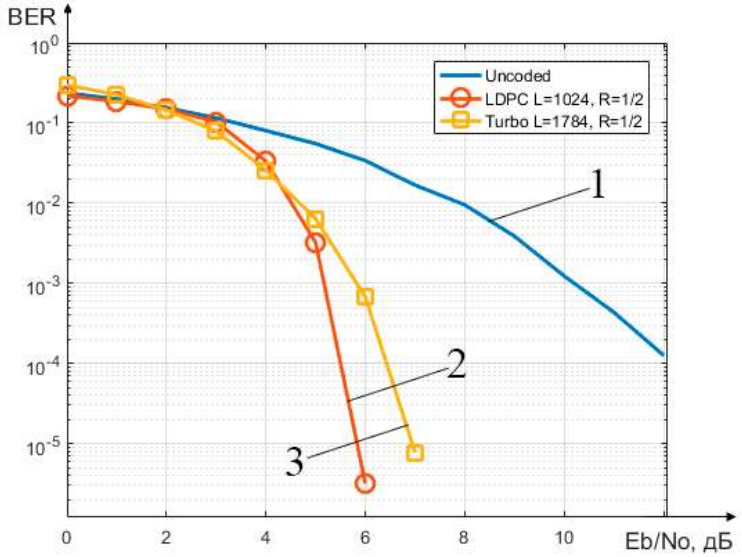

Рисунок 2.35 – Графік залежності імовірності бітової помилки від відношення сигнал/шум у випадку LDPC та турбо коду:
1 – некодоване;
2 – LDPC код для $L = 1024$, $R = 1/2$;
3 – турбо код для $L = 1784$, $R = 1/2$

На рисунку 2.35 видно, що LDPC код з найменшою із рекомендованих довжин, порівняно із турбо-кодом найменшої довжини, та швидкістю $1/2$ має невеликий енергетичний виграш і гарантує меншу імовірність бітової помилки на рівні відношення сигнал/шум 5 та 6 дБ. Хоча блок турбо-коду є більшим майже вдвічі, LDPC код в такому випадку застосовувати вигідніше.

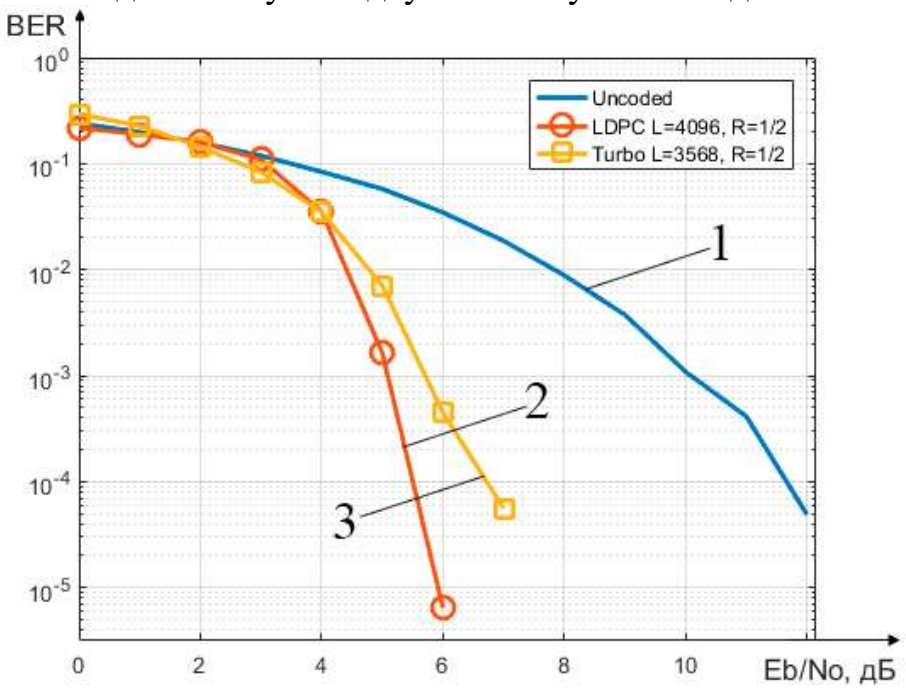

Рисунок 2.36 – Графік залежності імовірності бітової помилки від відношення сигнал/шум у випадку LDPC та турбо коду:
1 – некодоване;
2 – LDPC код для $L = 4096$, $R = 1/2$;
3 – турбо код для $L = 3568$, $R = 1/2$

У порівнянні LDPC коду та турбо-коду у випадку, коли коди мають більшу довжину блоку та швидкість кодів складає $1/2$, але тепер LDPC код має трохи більшу довжину, ніж турбо-код на рисунку 2.36, можна побачити, що LDPC код має ще переконливішу ефективність, порівняно з турбо-кодом. На відмітках відношення сигнал/шум в 6 та 7 дБ турбо-код має імовірність помилки після декодування, а LDPC майже позбавився її.

У подальшому математичному моделюванні та порівняльному аналізі використовувалися турбо-коди із швидкістю, яка складає $1/3$, але рекомендовані LDPC коди на мають такої або ще нижчої швидкості. Як вже було сказано, коди було наближено один до одного і от яким шляхом. Виходячи з того факту, що швидкість $1/2$ турбо-коду досягається за рахунок перфорації кожного другого перевірочного біту, тобто за першим інформаційним символом йде перевірочний символ з першого компонентного кодера, а за другим інформаційним символом йде перевірочний символ з другого компонентного кодера, і так по черзі, а на стороні приймача відсутні перевірочні символи заповнюються нулями перед декодуванням. Тобто, щоб отримати швидкість $1/3$ турбо-коду – не треба виколювати перевірочні біти, а до кожний інформаційний символ матиме два перевірочних символи з кожного компонентного кодера.. Такий самий метод

досягнення швидкості $1/2$ використовується і в LDPC кодах, але виколюється «хвіст» перевірочних символів, а на стороні приймача приклеюється «хвіст» тієї ж довжини, що було перфоровано, але повністю заповнений нулями. Отже було прийняте рішення не виколювати хвіст LDPC кодів, щоб знизити швидкість коду та підвищити його завадостійкість, а також наблизити швидкість до $1/3$, як і в турбо-кодах, але «хвіст» в LDPC кодах недостатньо довгий, щоб таким чином можна було сформувати швидкість $1/3$. В отриманих LDPC кодах швидкість складає $1/2.5$, і є трохи більшої, ніж в турбо-кодах.

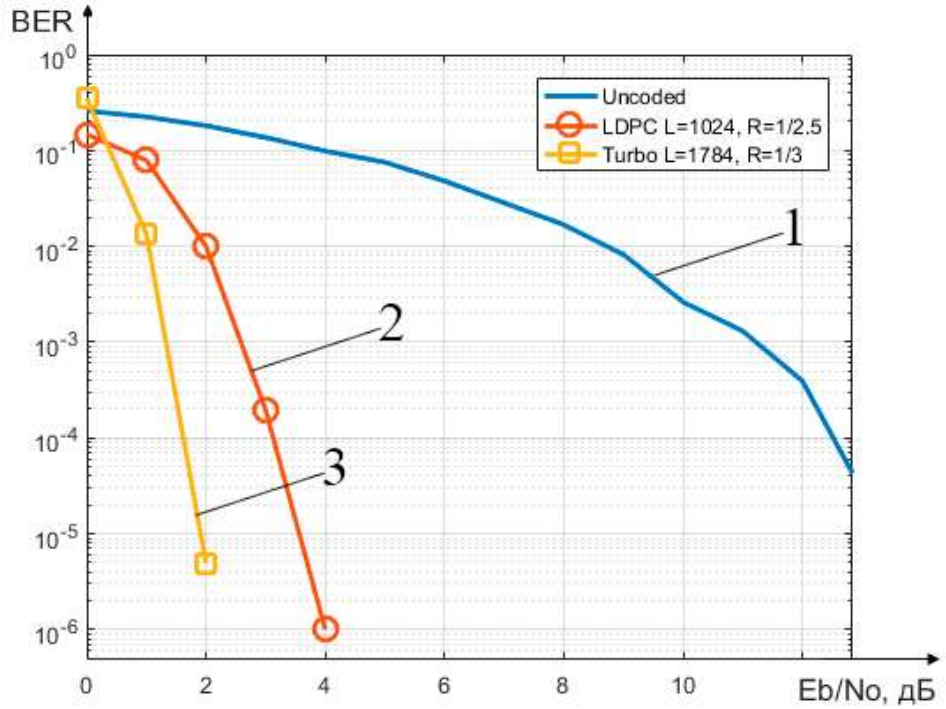

Рисунок 2.37 – Графік залежності імовірності бітової помилки від відношення сигнал/шум у випадку LDPC та турбо коду:
1 – некодоване;
2 – LDPC код для $L = 1024$, $R = 1/2.5$;
3 – турбо код для $L = 1784$, $R = 1/3$

На рисунку 2.37, при порівнянні кодів з найменшою довжиною інформаційного блоку та швидкістю $1/3$ турбо-коду і $1/2.5$ LDPC коду, було виявлено, що турбо-код із такими параметрами відчуває себе впевненіше, ніж LDPC код, але слід не забувати, що LDPC код має більшу швидкість. І навіть в такому вигляді завадостійкі коди гарантують відмінну ефективність, яка складає приблизно 8.9 дБ на величині імовірності бітової помилки 10^{-4}, порівняно із каналом без завадостійкого кодування.

На рисунку 2.38 показано графік залежності імовірності бітової помилки при порівнянні LDPC коду з довжиною блоку 4096 бітів і швидкістю $1/2$ та турбо-коду з довжиною блоку 1784 та швидкістю $1/3$. В теорії, чим більше довжина блоку, що передається, тим менше імовірність бітової помилки після декодування, і у випадку безкінечної довжини блоку, що передається – імовірність бітової помилки прагне до межі Шеннона. Але збільшення довжини

інформаційного блоку коду програє зменшенню швидкості за рахунок додавання перевірочних символів, що і можна побачити на рисунку 2.38.

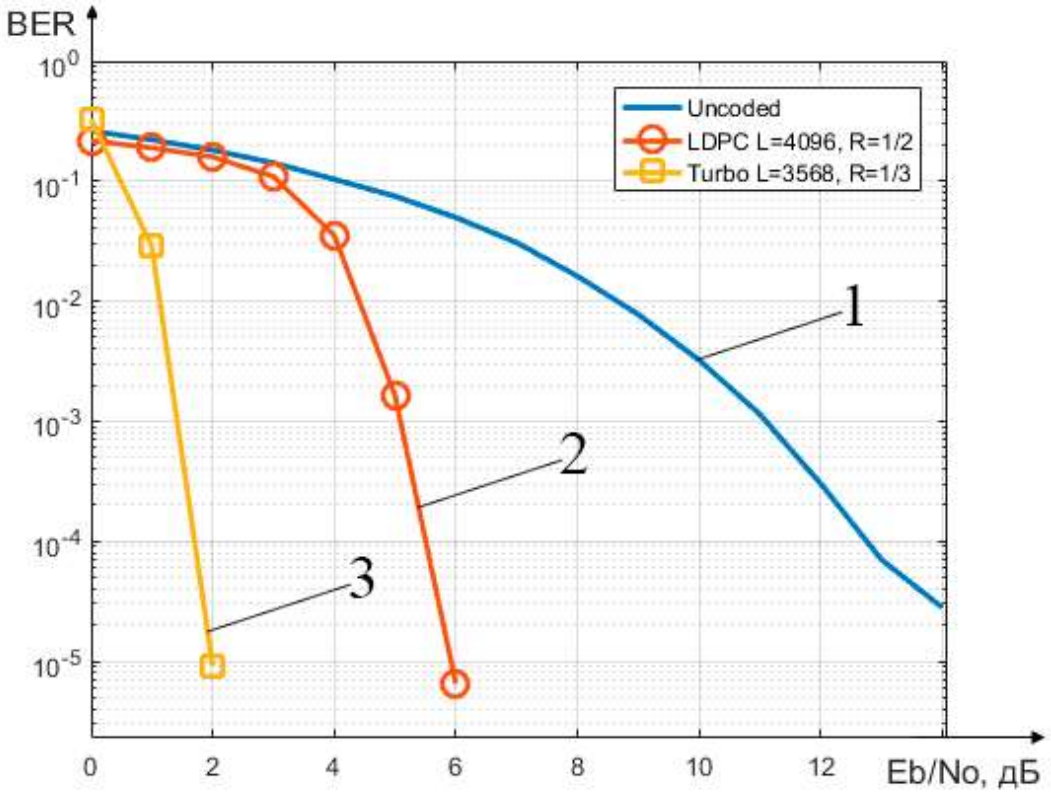

Рисунок 2.38 – Графік залежності імовірності бітової помилки від відношення сигнал/шум у випадку LDPC та турбо коду:
1 – некодоване;
2 – LDPC код для $L = 4096$, $R = 1/2$;
3 – турбо код для $L = 3568$, $R = 1/3$

В останньому випадку порівняння LDPC та турбо-коду було змодельовано LDPC код з довжиною 4096 бітів та швидкістю $1/2.5$, що в результаті дає кодове слово довжиною 10240 бітів, та турбо-код довжиною 3568 та швидкістю $1/3$, що дає кодове слово довжиною 10704, майже рівні випадки, але LDPC все ж таки має більший інформаційний блок і менше перевірочних символів. Результати математичного моделювання показано на рисунку 2.39.

Побудований спеціальним чином LDPC код має нижчу ефективність, але різниця між LDPC та турбо-кодом з меншою швидкістю зовсім мізерна.

Таким чином, в результаті аналізу двох запропонованих та рекомендованих до використання завадостійких кодів можна виділити декілька властивостей і нюансів використання того чи іншого.

При майже однаковій довжині інформаційного блоку і швидкості, яка складає $1/2$ LDPC код має декілька кращі показники ефективності.

У випадку коли коди не використовують перфорацію перевірочних бітів, турбо-код є кращим вибором, ніж LDPC.

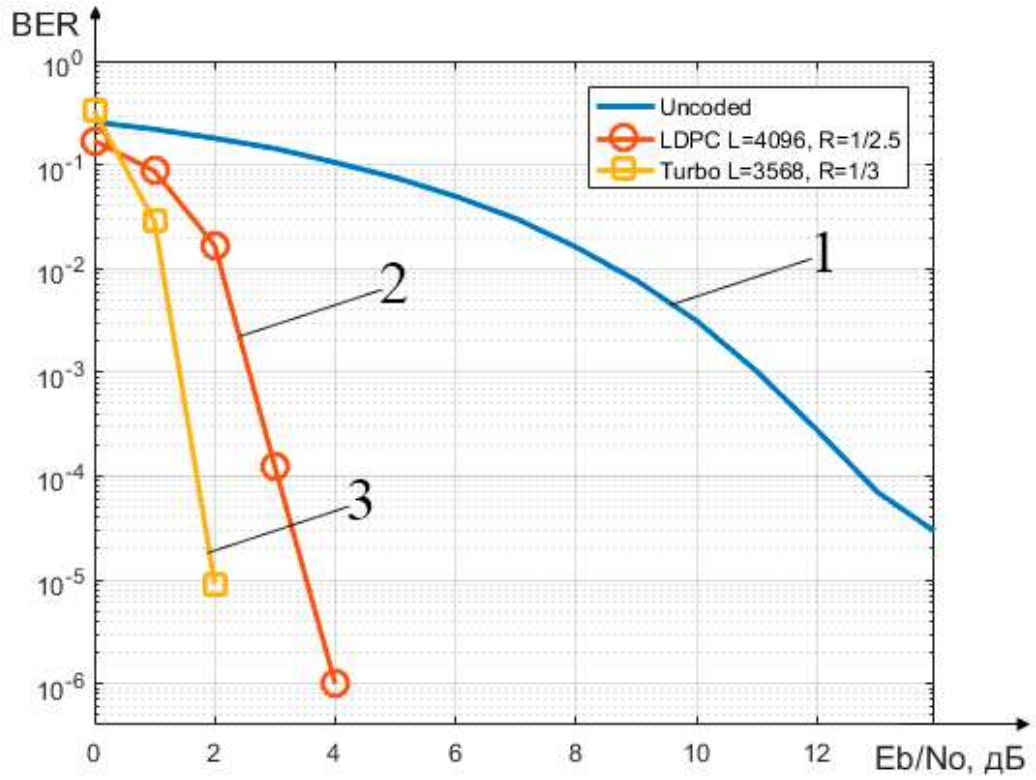

Рисунок 2.39 – Графік залежності імовірності бітової помилки від відношення сигнал/шум у випадку LDPC та турбо коду:
1 – некодоване;
2 – LDPC код для $L = 4096$, $R = 1/2.5$;
3 – турбо код для $L = 3568$, $R = 1/3$

Також слід відмітити, що турбо-коди мають більшу затримку, ніж LDPC коди, бо апріорна інформація надходить від одного компонентного декодера до іншого, чого позбавлені LDPC коди. Ще затримка вище при використанні турбо-кодів через декодування в декілька проходів по решітці, а у випадку LDPC кодів існує лише обчислення змінних та перевірочних вузлів в одному напрямку.

Ще однією особливістю кодів, які розглядаються є їх швидкості. Рекомендовані LDPC коди з сімейства AR4JA мають номінальну швидкість $1/2$ та інші: $2/3, 4/5, 7/8$. Найповільнішим кодом в цьому сімействі є код зі швидкістю $1/2$, а найшвидшим - $7/8$. У випадку ж турбо-кодів спостерігається протилежна картина. Рекомендований турбо-код із швидкістю $1/2$ – є найшвидшим, а код зі швидкістю $1/6$ – найповільнішим, але й дуже завадостійким. Таким чином в цих кодах спостерігаються різні підходи, умови застосування та відповідні результати їх роботи. LDPC коди прискорюються майже без втрати завадостійкості, а турбо-коди сповільнюються на кожному кроці гарантуючи все більшу завадостійкість.

3 ПРОГРАМНО-АПАРАТНА РЕАЛІЗАЦІЯ LDPC І ТУРБО КОДУВАННЯ

Для реалізації алгоритмів завадостійкого кодування в радіолініях зв'язку з космічними апаратами у фізичному вигляді використовуються спрощені версії цих самих алгоритмів тому, що на ПЛІС та мікроконтролерах досить складно здійснити операції ділення, багаточисельні операції добутку та інші. Такі операції на ПЛІС та мікроконтролерах виконують спеціальні блоки, які займають велику кількість ресурсів на пристрої і їх застосування у великій кількості неможливе на сьогоднішній момент. Для зменшення кількості ресурсів, які необхідні для реалізації алгоритмів завадостійкого кодування були створені спрощені версії, які несуттєво втрачають властивості завадостійкості, але потребують суттєво менше ресурсів на їх імплементацію та використання. Про такі спрощенні алгоритми декодування і піде мова в цьому розділі.

3.1 Спрощені алгоритми декодування LDPC-кодів

LDPC кодування, як вже було сказано в другому розділі, є операцією добутку інформаційної послідовності з генераторною матрицею обраного LDPC коду. Шукати добуток матриці на вектор на ПЛІС такого розміру – процес не з легких та ще й потребує величезних ресурсів, тому для кодування було рекомендовано пристрій кодування, який показано на рисунку 2.9 та розглянуто в другому розділі. Більший інтерес представляє спрощені алгоритми LDPC декодування.

3.1.1 Min-sum алгоритм декодування LDPC кодів

Фокус спрощених декодерів, які наближені до алгоритму суми добутків зосереджено на складному процесорі перевірочного вузла.

Розглядаючи рівняння (1.76) для $L_{i\to j}$ в логарифмічному декодері суми добутків звернемо увагу на форму $\phi(x)$, що є найбільшим доданком у сумі, який відповідає найменшому β_{ji}, тому можна припустити, що цей доданок домінує в сумі,

$$\phi\left(\sum_{j'}\phi\left(\beta_{j'i}\right)\right)\simeq\phi\left(\phi\left(\min_{\mathrm{j}'}\beta_{j'i}\right)\right)=\min_{\mathrm{j}'\in\mathrm{N(i)}-\{j\}}\beta_{j'i}\,. \tag{3.1}$$

Таким чином, min-sum алгоритм декодування – це просто алгоритм суми добутків із зміненим кроком оновлення перевірочного вузла на

$$L_{i\to j}=\prod_{j'\in N(i)-\{j\}}a_{j'i}\cdot\min_{j'\in N(i)-\{j\}}\beta_{j'i}\,. \tag{3.2}$$

Також можна показати, що у випадку каналу з адитивним білим Гаусовим шумом ініціалізація $L_{j\to i}=2y_j/\sigma^2$ може бути замінена на $L_{j\to i}=y_j$, коли використовується алгоритм min-sum[18], [19]. Звичайно, перевага полягає в тому, що оцінка потужності шуму σ^2 в цьому випадку непотрібна. На рисунку 3.1 показано приклад роботи такого алгоритму в двійковому каналі зі стираннями.

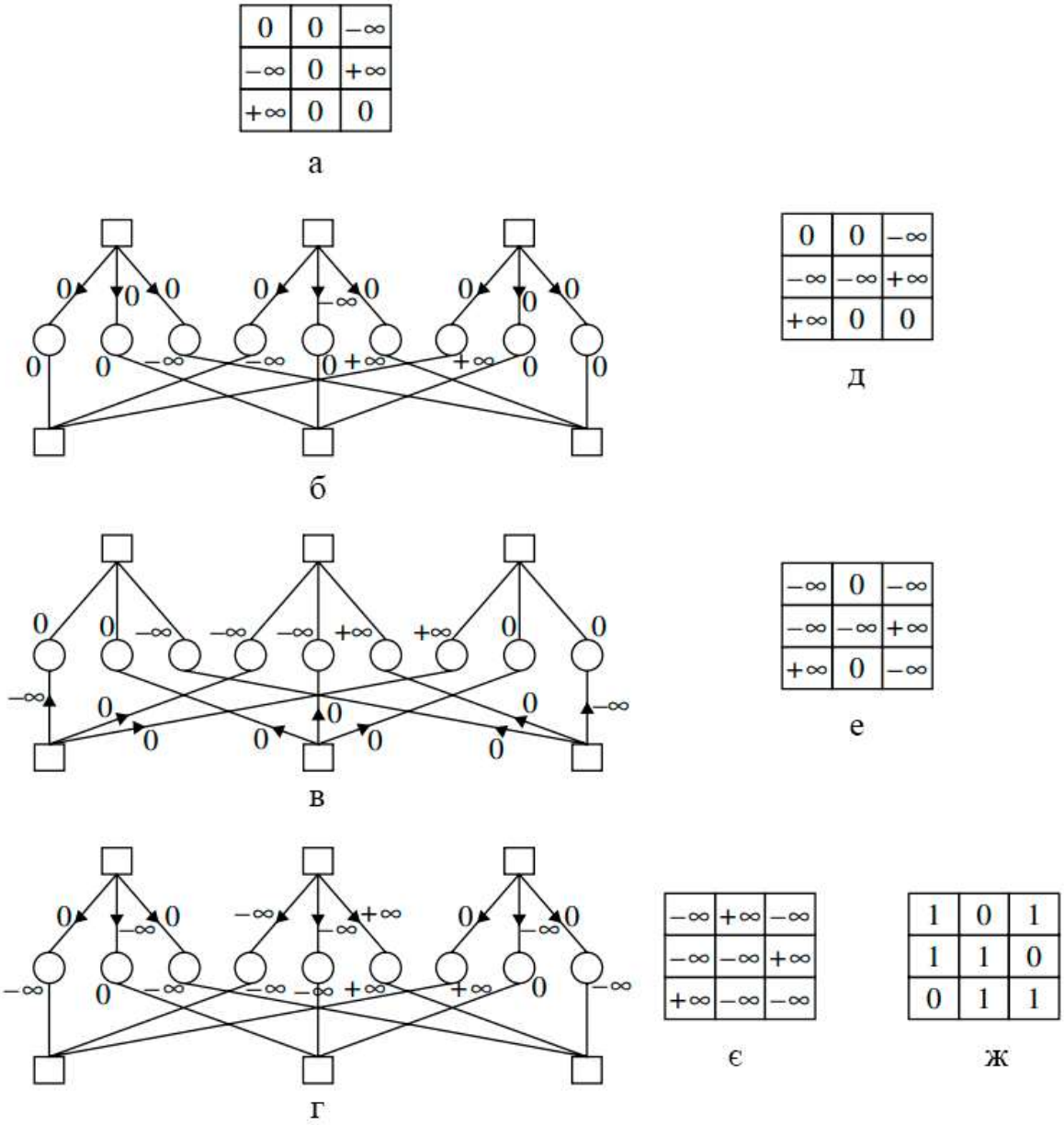

Рисунок 3.1 – Приклад роботи min-sum алгоритму декодування LDPC кодів:
а – отримані правдоподібності;
б – декодування по рядках;
в – декодування по стовбцях;
г – декодування по рядках;
д – оновлені правдоподібності після декодування по рядках;
е – оновлені правдоподібності після декодування по стовбцях;
є – оновлені правдоподібності після декодування по рядках;
ж – правдоподібності в бінарному вигляді

Ефективність квазициклічних кодів, які використовують алгоритм min-sum наведено в [16]. Також в [17] наведено техніку, яку слід застосовувати для таких кодів. Алгоритм min-sum з корекцією розглянуто в [20].

3.2 Спрощені алгоритми декодування турбо-кодів

Процес кодування турбо-кодів вже було розглянуто вище, тож перейдемо до спрощених алгоритмів декодування кодових блоків, отриманих за допомогою турбо-кодів.

3.2.1 Log-MAP алгоритм декодування турбо-кодів

Недоліком BCJR алгоритму є те, що необхідно виконувати велику кількість множень. Перехід до логарифмічного представлення дозволяє замінити операцію добутку сумою та зменшити складність алгоритму за тієї ж точності та продуктивності. В логарифмічному вигляді вихідні значення алгоритму Log-MAP приймають наступний вигляд

$$L(\hat{u}_i) = \log\frac{\Pr(\hat{u}_i = 1|r)}{\Pr(\hat{u}_i = 0|r)} = \log\left(\frac{\sum_{(p,q)\in Q_1}\alpha_i(p)\gamma_i(p,q)\beta_{i+1}(q)}{\sum_{(p,q)\in Q_0}\alpha_i(p)\gamma_i(p,q)\beta_{i+1}(q)}\right). \quad (3.3)$$

Якщо визначити $\log\big(\alpha_i(p)\big)$, $\log\big(\beta_{i+1}(q)\big)$, $\log\big(\gamma_i(p,q)\big)$ як $A_i(p)$, $B_{i+1}(q)$ та $\Gamma_i(p,q)$ відповідно, то вираз (3.3) можна записати як

$$L(\hat{u}_i) = \log\left(\sum_{(p,q)\in Q_1}\exp\big(A_i(p) + \Gamma_i(p,q) + B_{i+1}(q)\big)\right) - \\ - \log\left(\sum_{(p,q)\in 0}\exp\big(A_i(p) + \Gamma_i(p,q) + B_{i+1}(q)\big)\right). \quad (3.4)$$

Обчислення метрик $A_{i+1}(q)$ та $B_i(p)$ буде виглядати наступним чином

$$A_{i+1}(q) = \log\left(\sum_{p=0}^{Q-1}\exp\big(A_i(p) + \Gamma_i(p,q)\big)\right), \quad (3.5)$$

$$B_i(p) = \log\left(\sum_{q=0}^{Q-1}\exp\big(B_{i+1}(q) + \Gamma_i(p,q)\big)\right). \quad (3.6)$$

Для того, щоб розрахувати метрику переходу в логарифмічній області $\Gamma_i(p,q)$ скористаємося виразом (1.114), тоді

$$\Gamma_i(p,q) = -c\log\big(\sqrt{2\pi}\sigma\big) - \frac{1}{2\sigma^2}\left\|r_i - \hat{a}_i^{(p,q)}\right\|^2 + \log(\Pr(u_i = x)). \quad (3.7)$$

При обчисленні метрики ребра постійною $-c\log\big(\sqrt{2\pi}\sigma\big)$ можна знехтувати, так як вона скорочується при обчисленні остаточної надійності біта.

3.2.2 Max-Log-MAP алгоритм декодування турбо-кодів

Тим не менш Log-MAP є обчислювально складним і потребує обчислення експонент та логарифмів. Для зменшення складності декодера на практиці часто користуються наближеним способом розрахунку метрик станів та шляхів, за рахунок приблизного обчислення логарифму суми експонент:

$$\log\left(\sum_i \exp\left(x^i\right)\right) \approx \max_{\mathrm{i}}(x_i)\,. \tag{3.8}$$

Алгоритм, який використовує це правило для розрахунку метрик, називається Max-Log-MAP. Розрахунок метрик $A_{i+1}(q)$ та $B_i(p)$ зводиться до наступних нескладних правил

$$A_{i+1}(q) \approx \max_{\mathrm{p}}\left(A_i(p) + \Gamma_i(p,q)\right), \tag{3.9}$$

$$B_i(p) \approx \max_{\mathrm{q}}\left(B_{i+1}(q) + \Gamma_i(p,q)\right). \tag{3.10}$$

Під час обчислення апостеріорних імовірностей біт також можна скористатися наближенням (3.8), тоді

$$L(\hat{u}_i) = \max_{(p,q)\in Q_1}\left(A_i(p) + \Gamma_i(p,q) + B_{i+1}(q)\right) - \\ - \max_{(p,q)\in Q_0}\left(A_i(p) + \Gamma_i(p,q) + B_{i+1}(q)\right). \tag{3.11}$$

Перевагою Max-Log-MAP алгоритму є його невелика обчислювальна складність. Також слід відмітити, що для роботи Log-MAP алгоритму необхідно враховувати значення дисперсії каналу з адитивним білим Гаусовим шумом. Для Max-Log-MAP алгоритму це неважливо, так як для обчислення значень всіх метрик використовується лише операція суми і коефіцієнтом L_c можна знехтувати. Проте обчислені апостеріорні імовірності біт є лише наближеними значеннями, що негативно позначається при декодуванні турбо-коду. Для уточнення обчислених значень надійності у виразі (3.8) можна використати корегуючий коефіцієнт, тоді вираз (3.8) приймає наступний вигляд

$$\log(\exp(x_1) + \exp(x_2)) = \max(x_1, x_2) + f(x_1, x_2) \tag{3.12}$$

Уперше цей метод було запропоновано П. Робертсоном та іншими в [27]. Функція $f(x_1, x_2)$ є корекцією, і якщо вона дорівнює $\log(1 + exp(-|x_1 + x_2|))$, то це еквівалентно використанню Log-MAP алгоритму, і дозволяє замість обчислення двох експонент обчислювати лише одну. Проте, для того, щоб не обчислювати значення експонент взагалі, автори пропонують зберігати в таблиці невелику кількість корегуючих коефіцієнтів, які обираються в залежності від конкретних значень x_1 та x_2.

4 ЕКОНОМІЧНА ЧАСТИНА. РОЗРАХУНОК СОБІВАРТОСТІ ПРОГРАМНОГО ПРОДУКТУ

4.1 Мета економічного розділу

В цьому розділі дипломної роботи виконується розрахунок затрат на розробку програмного продукту, який являє собою алгоритм математичного моделювання.

Метою даного розділу є розрахунок:

- трудомісткості виконання робіт;
- заробітної плати;
- амортизації основних засобів;
- визначення собівартості математичного моделювання.

4.2 Розрахунок собівартості математичного моделювання

Для ведення проекту в цілому та керівництвом ходом роботи необхідна посада керівника. Для розробки алгоритму, програми та її тестування, а також звітування про працездатність моделювання, а за тим налаштування та введення в експлуатацію необхідна участь розробника. Також розглянемо випадок, коли цю роботу виконуватимуть керівник та студент-практикант.

Наведемо перелік робіт для розробників програмного продукту. Було встановлено тривалість робіт для кожного із співробітників, а також тривалість роботи над проектом в цілому.

Перелік робіт виконуваних співробітниками та їх тривалість наведені в таблиці 4.3 і таблиці 4.4 та становить 20 робочих днів у місяці.

В таблиці 4.1 і таблиці 4.2 приведені посади співробітників, їх місячні та денні оклади в гривнях.

Таблиця 4.1 – Склад виконавців проекту (перший варіант)

Посади	Посадові оклади, грн	
	Місячні	Денні
Керівник	22000	1100
Програміст	16000	800

Таблиця 4.2 – Склад виконавців проекту (другий варіант)

Посади	Посадові оклади, грн	
	Місячні	Денні
Керівник	22000	1100
Студент-практикант	9500	475

Таблиця 4.3 – Трудомісткість робіт (перший варіант)

Вид робіт	Тривалість, дні	Трудомісткість, люд-дні	Виконавець	
			Керівник проекту	Програміст
Попередня робота				
Постановка задачі	4	8	+	+
Аналіз предметної області	7	14	+	+
Розробка технічного завдання (ТЗ)				
Розробка вимог до алгоритму	2	4	+	+
Погодження та затвердження ТЗ	3	6	+	+
Розробка програмного продукту				
Розробка алгоритму	15	30	+	+
Розробка програми	14	14		+
Тестування	7	7		+
Розробка звітів про працездатність	5	5		+
Впровадження програмного продукту				
Випробовування програмного продукту	10	10		+
Здача математичного моделювання	3	6	+	+
Разом	70	101	34	70

Таблиця 4.4 – Трудомісткість робіт (другий варіант)

Вид робіт	Тривалість, дні	Трудомісткість, люд-дні	Виконавець	
			Керівник проекту	Програміст
Попередня робота				
Постановка задачі	6	6	+	

Продовження таблиці 4.4

Вид робіт	Тривалість, дні	Трудомісткість, люд-дні	Виконавець	
			Керівник проекту	Програміст
Аналіз предметної області	7	14	+	+
Розробка технічного завдання (ТЗ)				
Розробка вимог до алгоритму	3	3	+	
Погодження та затвердження ТЗ	4	8	+	+
Розробка програмного продукту				
Розробка алгоритму	20	20	+	
Розробка програми	15	15		+
Тестування	7	7		+
Розробка звітів про працездатність	5	5		+
Впровадження програмного продукту				
Випробовування програмного продукту	11	22	+	+
Здача математичного моделювання	4	8	+	+
Разом	82	108	55	53

Далі необхідно обчислити фонд основної заробітної плати (ОЗП), який складається з заробітної плати керівника та програміста, помноженої на відповідну їм трудомісткість:

$$ОЗП = ЗП_{дк} \cdot Т_к + ЗП_{дп} \cdot Т_п, \quad (4.1)$$

де $ЗП_{дк}$ – денна заробітна плата керівника, грн;

$ЗП_{дп}$ – денна заробітна плата програміста або студента-практиканта, грн;

$Т_к$ – трудомісткість керівника, люд-дні;

$Т_п$ – трудомісткість програміста або студента-практиканта, люд-дні.

Таким чином, фонд основної заробітної плати, розрахований за формулою (4.1) дорівнює:

$$ОЗП_1 = 1100 \cdot 34 + 800 \cdot 70 = 37400 + 56000 = 93400 \text{ грн}, \quad (4.2)$$

$$ОЗП_2 = 1100 \cdot 55 + 475 \cdot 53 = 60500 + 26125 = 86625 \text{ грн}. \quad (4.3)$$

Тепер необхідно провести розрахунок фонду додаткової заробітної плати. ДЗП становить 18% від фонду основної заробітної плати і розраховується так:

$$ДЗП_1 = ОЗП \cdot 18\% = 93400 \cdot 0.18 = 16812 \text{ грн}. \quad (4.4)$$

$$ДЗП_2 = ОЗП \cdot 18\% = 86625 \cdot 0.18 = 15592 \text{ грн}. \quad (4.5)$$

Разом, фонд заробітної плати становить:

$$ФЗП_1 = ОЗП_1 + ДЗП_1 = 93400 + 16812 = 110212 \text{ грн}, \quad (4.6)$$

$$ФЗП_2 = ОЗП_2 + ДЗП_2 = 86625 + 15592 = 102217 \text{ грн}. \quad (4.7)$$

Нарахування на заробітну плату (Єдиний соціальний внесок – $Н_{ЄСВ}$) розраховується як 22% від фонду заробітної плати:

$$ФЗП_1 \cdot Н_{ЄСВ} = 110212 \cdot 0.22 = 24246 \text{ грн}, \quad (4.8)$$

$$ФЗП_2 \cdot Н_{ЄСВ} = 102217 \cdot 0.22 = 22487 \text{ грн}. \quad (4.9)$$

Далі необхідно розрахувати вартість матеріалів, комплектуючих та покупних виробів, необхідних для реалізації алгоритму, які перераховані в таблиці 4.5.

Таблиця 4.5 – Матеріали та покупні вироби (перший варіант)

Матеріали	Кількість, шт	Ціна, грн	Вартість, грн
Matlab (DSP System Toolbox), місячна ліцензія	3	2500	7500
Чорнила для принтеру	1	200	200
Упаковка паперу (100 аркушів)	3	150	450
Місячний пакет інтернету	3	200	600
Флеш-карта	2	300	600
Набір маркерів	1	100	100
Разом			9450

Таблиця 4.6 – Основні засоби (перший варіант)

Основні засоби	Кількість, шт	Ціна, грн	Вартість, грн
Персональний комп'ютер	2	30000	60000
Робоче місце	2	7000	14000
Принтер	1	6200	6200
Дошка магнітно-маркерна	1	3000	3000
Разом			83200

Таблиця 4.7 – Основні засоби (другий варіант)

Основні засоби	Кількість, шт	Ціна, грн	Вартість, грн
Персональний комп'ютер	2	22500	45000
Робоче місце	2	6800	13600
Принтер	1	6200	6200
Дошка магнітно-маркерна	1	3000	3000
Разом			67800

Амортизація – поступовий знос основних засобів (обладнання, будівлі, споруди) і перенесення їх вартості на продукт. Амортизацію обладнання як складову собівартості продукції пропонується розраховувати за такою формулою:

$$А_{м1} = \frac{ОЗ_1 \cdot 0.25 \cdot Д_{Р1}}{Д_Г} = \frac{83200 \cdot 0.25 \cdot 70}{240} = \frac{1456000}{240} = 6066 \text{ грн}, \quad (4.10)$$

$$А_{м2} = \frac{ОЗ_2 \cdot 0.25 \cdot Д_{Р2}}{Д_Г} = \frac{67800 \cdot 0.25 \cdot 82}{240} = \frac{1389900}{240} = 5791 \text{ грн}, \quad (4.11)$$

де $А_м$ – річні амортизаційні відрахування по обладнанню, грн;

$Д_Р$ – кількість днів розробки програмного продукту;

$Д_Г$ – кількість робочих днів у році.

Для виконання роботи необхідне приміщення, яке було орендоване. Вартість оренди офісного приміщення площею 30 м^2 для двох робочих місць становить 400 грн на день. Розрахуємо загальну вартість оренди на період проекту:

$$\text{Оренда}_1 = \text{Оренда в день} \cdot \text{Період проекту}_1 = 400 \cdot 70 = 28000 \text{ грн}, \quad (4.12)$$

$$\text{Оренда}_2 = \text{Оренда в день} \cdot \text{Період проекту}_2 = 400 \cdot 82 = 32800 \text{ грн}. \quad (4.13)$$

Розрахунок собівартості для першого варіанту наведено в таблиці 4.8, а для другого варіанту в таблиці 4.9

Таблиця 4.8 – Розрахунок собівартості (перший варіант)

№	Статті	Сума, грн	Примітка
1	Фонд основної заробітної плати (ОЗП)	93400	$ОЗП_1$
2	Додаткова заробітна плата	16812	$ДЗП_1$ (18% від $ОЗП_1$)
3	Єдиний соціальний внесок	24246	22% від ($ОЗП_1 + ДЗП_1$)
4	Матеріали та покупні вироби	9450	Табл. 4.5
5	Невиробничі витрати (НВ)	37360	40% від $ОЗП_1$
6	Оренда приміщення	28000	Оренда
7	Амортизація	6066	25% від $ОЗ_1 \cdot (\frac{Др}{Дг})$
8	Собівартість	215334	П1+П2+П3+П4+П5+П6+П7

Таблиця 4.9 – Розрахунок собівартості (другий варіант)

№	Статті	Сума, грн	Примітка
1	Фонд основної заробітної плати (ОЗП)	86625	$ОЗП_2$
2	Додаткова заробітна плата	15592	$ДЗП_2$ (18% від $ОЗП_2$)
3	Єдиний соціальний внесок	22487	22% від ($ОЗП_2 + ДЗП_2$)
4	Матеріали та покупні вироби	9450	Табл. 4.5
5	Невиробничі витрати (НВ)	34650	40% від $ОЗП_2$
6	Оренда приміщення	32800	Оренда
7	Амортизація	5791	25% від $ОЗ_2 \cdot (\frac{Др}{Дг})$
8	Собівартість	207395	П1+П2+П3+П4+П5+П6+П7

Висновки до економічного розділу: у цьому розділі були проведені розрахунки витрат, що необхідні для розробки програмного продукту, тобто були обчислені трудомісткості двох посад в двох варіантах: керівника та програміста, і керівника і студента-практиканта які склали 34 і 70 люд-днів відповідно для першого варіанту, та 55 і 53 для другого варінту. За цим були розраховані, по відповідним денним окладам, основна та додаткова заробітна плата, яка склала для першого варіанту: 93400 і 16812 гривень відповідно, та для другого варіанту: 86625 і 15592 гривень відповідно. Після того як була складена таблиця основних засобів – розрахована амортизація основного обладнання, яка склала для першого варіанту 6066 гривень, а для другого варіанту 5791 гривень. В результаті, собівартість розробки програмного продукту складає 215334

гривень – для першого варіанту, та 207395 гривень – для другого варіанту. Таким чином, хоча варіант зі студентом-практикантом займає більшу кількість днів та людо-днів розробки, що тягне за собою більші витрати на оренду, у підсумку такий варіант має меншу собівартість розробки програмного продукту. Результат даної роботи не підлягає продажу, а необхідний для використання у лабораторному практикумі, тому в даному розділі не розглядається ціна, а лише собівартість продукту.

ВИСНОВОК

Таким чином, в роботі були розглянуті моменти які безпосередньо стосуються радіоліній зв’язку з космічними апаратами. Зачіпаються види місій в радіолініях зв’язку з космічними апаратами, рекомендовані методи обробки сигналів в цих радіолініях, їх основні параметри та частотні діапазони, а також рекомендовані методи модуляції, які використовуються в таких радіолініях, а саме BPSK, QPSK, 8-PSK та M-APSK види модуляції. Велику увагу приділено завадостійкому кодуванню інформації, розгляд включає в себе опис методів кодування та декодування турбо та LDPC кодів, які рекомендовані до використання в радіолініях зв’язку з космічними апаратами.

При виконанні роботи були побудовані математичні моделі симуляцій двох завадостійких кодеків з різними параметрами та довжинами інформаційного блоку для передачі. Симуляція вміщувала в себе кодування псевдовипадкової інформаційної послідовності двома видами завадостійкого кодування, LDPC та турбо, передачі її через канал з адитивним білим Гаусовим шумом та декодування за допомогою декодерів, працюючих за алгоритмами з максимумом апостеріорної імовірності. В результаті великої кількості симуляцій з різноманітними випадками довжин, швидкостей та умов декодування були отримані графіки залежності імовірності бітової помилки від відношення сигнал/шум повідомлень із застосуванням завадостійкого кодування та без нього. Залежності імовірності бітової помилки після LDPC та турбо кодування було порівняно та проаналізовано в другому розділі. Можна сказати, що обидва види кодування гарантують відмінну ефективність, найкращу на сьогоднішній день, але кожен у своєму випадку краще іншого. Обрані для порівняння види завадостійкого кодування більше ніж інші наближаються до межі Шеннона, під якою розуміється максимальна швидкість передачі, для котрої є можливість виправити помилки із заданим відношенням сигнал/шум, або мінімальне відношення сигнал/шум, для якого теоретично можлива безпомилкова передача та декодування блоку із заданою швидкістю.

Також були розглянуті шляхи спрощення алгоритмів LDPC та турбо декодування для їх програмно-апаратної реалізації та практичного застосування на програмованих логічних інтегральних схемах.

Результати роботи пропонується використовувати в навчальному процесі або реальних проектах для реалізації завадостійкого кодування в радіолініях зв’язку з космічними апаратами.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Бондаренко І. М. Системи радіозв'язку. Кн.2, ч.1. Радіолінії зв'язку: Навч. посібник. – Харків.: ХІ ВПС, 2003. – 162с.
2. Варакин Л. Е. Системы связи с шумоподобными сигналами: Уч.пособ. – Москва: Радио и связь, 1985. – 384с.
3. Кологривов В. Н. Эффект Доплера в классической физике:Уч. Пособ. – Москва: МФТИ, 2012. – 32с.
4. Регламент радиосвязи: ITU ВКР-12. – Действующий от 01.01.13. – G.:International Telecommunication Union, 2012. – 435с.
5. Прокіс Д. Цифровий зв'язок: Пер. з англ. – М.: Радіо та зв'язок, 2000. – 800с.
6. Скляр Б. Цифровий зв'язок. Теоретичні основи та практичне застосування: Пер. з англ. – М.: Вільямс, 2003. – 1104с.
7. Digital video: Science book / University Campus STeP Ri; compose by K. P. Liolis, R. De Gaudenzi, N. Alagha, A. Martinez, A. G. i Fàbregas. – C., 2010. – 500 p.
8. R. De Gaudenzi, A. G. Fabregas, A. Martinez Performance analysis of turbocoded APSK modulations over nonlinear satellite channels // IEEE transactions on wireless communications. – 2006. – Vol. 5, № 9, – P. 2396 – 2407.
9. R. G. Gallager Low-Density Parity-Check Codes: Diss… phd. – C., 1963. – 90 p.
10. R. M. Tanner A recursive approach to low complexity codes // IEEE Trans. Information Theory. – 1981. – Vol. 27, № 9, P. 533–547.
11. D. MacKay, R. Neal, Good codes based on very sparse matrices // Cryptography and Coding, 5th IMA Conf.: Material of science conference. – B., 1995. – P. 1 – 13.
12. D. MacKay Good error correcting codes based on very sparse matrices // IEEE Trans. Information Theory. – 1999. – Vol. 45, № 3, P. 399–431.
13. Y. Wang and M. Fossorier Doubly generalized LDPC codes // IEEE Trans. Information Theory. – 2006. – Vol. 57, № 5, P. 669–673.
14. J. Pearl Probabilistic Reasoning in Intelligent Systems: Science book. – San Francisco: Morgan Kaufmann Publishers, Inc, 1988. – 573 p.
15. F. Kschischang, B. Frey, and H.-A. Loeliger, "Factor graphs and the sum-product algorithm // IEEE Trans. Information Theory. – 2001. – Vol. 47, № 2, P. 498–519.
16. Y. Zhang Design of Low-Floor Quasi-Cyclic IRA Codes and Their FPGA Decoders: Diss… phd. – T., 2007. – 128 p.
17. Y. Han and W. E. Ryan Low-floor decoders for LDPC codes // IEEE Trans. Communications. – 2009. – Vol. 57, № 6 P. 1663–1673.
18. J. Zhao, F. Zarkeshvari, and A. Banihashemi On implementation of min-sum algorithm and its modifications for decoding low-density parity-check

(LDPC) codes // IEEE Trans. Communications. – 2005. – Vol. 53, № 4, P. 549–554.
19. J. Chen, A. Dholakia, E. Eleftheriou, M. Fossorier, and X.-Y. Hu Reduced-complexity decoding of LDPC codes // IEEE Trans. Communications. – 2005. – Vol. 53, № 8, P. 1288–1299.
20. X.-Y. Hu, Regular and Irregular Progressive Edge-Growth Tanner Graph // IEEE Trans. On Information Theory. – 2005. – Vol. 51, № 1, P. 386 – 398.
21. C. Jones, E. Valles, M. Smith, and J. Villasenor Approximate-min* constraint node updating for LDPC code decoding // IEEE Military Communications Conf: Material of science conference. – LA., P. 157–162.
22. L. Bahl, J. Cocke, F. Jelinek, and J. Raviv Optimal decoding of linear codes for minimizing symbol error rate // IEEE Trans. Information Theory. – 1974. – Vol. 20, № 3, P. 284–287.
23. R. McEliece On the BCJR trellis for linear block codes // IEEE Trans. Information Theory. – 1996. – Vol. 42, № 7, P. 1072–1092.
24. C. Berrou, A. Glavieux, P. Thitimajshima Near Shannon limit error-correcting coding: turbo codes // Proceedings of the IEEE International Conference on Communications (ICC 93): Material of science conference. – P., 1993. – Vol. 2. – P. 1064–1070.
25. L. Bahl, J. Cocke, F. Jelinek, J. Raviv Optimal decoding of linear codes for minimizing symbol error rate // IEEE Transactions on Information Theory. – 1974. – Vol. 20, № 2. – P. 284–287.
26. J. Hagenauer, P. Hoeher. A viterbi algorithm with soft-decision outputs and its applications // Proc. IEEE GLOBECOM ’89: Material of science conference. – 1989. – Vol. 3. – P. 1680–1686.
27. P. Robertson, E. Villebrun, P. Hoeher comparison of optimal and sub-optimal MAP decoding algorithms operating in the log domain // Proceedings of the IEEE International Conference on Communications (ICC 95): Material of science conference. – 1995. – Vol. 2. – P. 1009–1013.
28. D. Divsalar, F. Pollara On the design of turbo codes // The Telecommunications and Data Acquisition Progress Report, TDA PR 42-123: Material of science conference. – 1995. – P. 99–121.
29. D. Divsalar, S. Dolinar Weight distributions for turbo codes using random and nonrandom permutations // The Telecommunications and Data Acquisition Progress Report, TDA PR 42-122: Material of science conference. – 1995. – P. 56–65.
30. В.В. Зяблов, М.А. Цветков. Дистанционные свойства турбо кодов с различными перемежителями // Информационные процессы. – 2003. – Т. 3, № 2. — С. 83–96.
31. Chatzigeorgiou, I. A. Performance Analysis and Design of Punctured Turbo Codes: Diss… phd. C., 2006. – 150 p.
32. Divsalar, F. Pollara Multiple turbo codes for deep-space communications // PL TDA Progress Report 42-121: Material of science conference. – 1995. – P. 66–77.

33. A. Heim, U. Sorger Turbo decoding: Why stopping-criteria do work // 5th International Symposium on Turbo Codes and Related Topics: Material of science conference. – 2008. – P. 255–259.
34. T. K. Moon Error Correction Coding. Mathematical Methods and Algorithms: Science book. – New Jersey: John Wiley and Sons, Inc., 2005. — P. 800.
35. Brink S. Convergence behavior of iteratively decoded parallel concatenated codes // IEEE Transactions on Communications. – 2001. – Vol. 49, № 10. – P. 1727–1737.
36. A. Matache, S. Dolinar, F. Pollara Stopping rules for turbo decoders // TMO Progress Report 42-142: Material of science conference. – 2000. – P. 1–22.
37. F. Gilbert, F. Kienle, N. Wehn Low complexity stopping criteria for UMTS turbo decoders // The 57'th IEEE Semiannual Vehicular Technology Conference: Material of science conference. – 2003. Vol. 4. – P. 2376–2380.
38. J. Zhang, Y. Wang, M. P. C. Fossorier, J.S. Yedidia Replica shuffled iterative decoding // Proceedings of the 2005 International Symposium on Information Theory: Material of science conference. – 2005. – P. 454–458.
39. J. Zhang, M. P. C. Fossorier, J.S. Yedidia Iterative decoding with replicas // IEEE Transactions on Information Theory. – 2007. – Vol. 53, № 5. – P. 1644–1663.
40. Viterbi, A. J. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm // IEEE Transactions on Information Theory. — 1967. — Vol. 13, № 2. — P. 260–269.
41. G. D. Forney Jr. The Viterbi algorithm // Proceedings of the IEEE. — 1973. — Vol. 61, № 3. — P. 268–278.
42. Kamuf M. Trellis Decoding. From Algorithm to Flexible Architectures: Science book. – Lund: Tryckeriet i E-huset, 2007. — P. 128.
43. M. P. C. Fossorier, S. Lin, C. Xu Bi-directional SOVA decoding for turbo-codes // IEEE Communications Letters. — 2000. — Vol. 4, № 12. — P. 405–407.
44. L. Hanzo and others. Turbo Coding, Turbo Equalisation and Space-Time Coding for Transmission over Fading Channels: Science book / L. Hanzo T.H. Liew, B.L. Yeap; University of Southampton. – S., 2002. –209 p.
45. Johnson S. J. Iterative Error Correction. Turbo, Low-Density Parity-Check and Repeat–Accumulate Codes: Science book. – Newcastle: Cambridge University Press, 2009. – 331 p.
46. R. Garello, F. Chiaraluce, P. Pierleoni et al. On error floor and free distance of turbo codes // IEEE International Conference on Communications (ICC 2001): Material of science conference. – 2001. – Vol. 1. – P. 45–49.
47. T. Richardson The geometry of turbo-decoding dynamics // IEEE Transactions on Information Theory. – 2000. — Vol. 46, № 1. – P. 9–23.
48. J. Hagenauer, E. Offer, L. Papke Iterative decoding of binary block and convolutional codes // IEEE Transactions on Information Theory. – 1996. – Vol. 42, № 2. – P. 429–445.

49. J. Zhang, Y. Wang, M. P. C. Fossorier, J.S. Yedidia Reduced latency turbo decoding // IEEE 6th Workshop on Signal Processing Advances in Wireless Communications: Material of science work, 2005. – P. 930–934.
50. D. Divsalar, S. Dolinar, C. Jones Low-Rate LDPC Codes with Simple Protograph Structure // In Proceedings of the IEEE International Symposium on Information Theory: Material of science conference. – A., 2005. – P. 1622–1626.
51. TM synchronization and channel coding: 131.0-B-3. – Active 01.09.2017. – W.: Recommended standard USA, 2017. – 97 p.