Міністерство освіти і науки України

Український державний університет науки і технологій

Факультет «Комп'ютерні технології і системи»

(назва факультету)

Кафедра «Електронні обчислювальні машини»

(повна назва кафедри)

Пояснювальна записка

до кваліфікаційної роботи

магістра

(ступінь вищої освіти)

на тему: Дослідження та розробка засобів вивчення решіткового кодування.___

за освітньою програмою Комп'ютерна інженерія

зі спеціальності: 123 Комп'ютерна інженерія

(шифр і назва спеціальності)

Виконав: студент групи: КС2121

	(підпис студента)	Вячеслав НІКІТІН (Ім'я ПРІЗВИЩЕ)
Керівник:	(підпис)	професор Ігор ЖУКОВИЦЬКИЙ (посада, Ім'я ПРІЗВИЩЕ)
Нормоконтролер:	(підпис)	доцент Володимир ШАПОВАЛОВ (посада, Ім'я ПРІЗВИЩЕ)

Засвідчую, що у цій роботі немає запозичень з праць інших авторів без відповідних посилань.

Студент _____ (підпис)

Дніпро – 2022 рік

Ministry of Education and Science of Ukraine

Ukrainian State University of Science and Technologies

Faculty «Computer technologies and systems»

(faculty)

Department «Electronic computers»

(department)

Explanatory Note

to Qualification Work

Master's

(higher education degree)

on the topic: Research and development of tools for studying lattice codding.

according to educational curriculum Computer Engineering

in the Speciality: 123 Computer Engineering

(speciality and its code)

Done by the student of the group: KC2121 Vyacheslav Nikitin

(name, surname)

Scientific Supervisor: Professor Igor Zhukovitskiy

(position, name, surname)

Normative controller : docent Volodymyr Shapovalov

(position, name, surname)

Dnipro – 2022

Міністерство освіти і науки України

Український державний університет науки і технологій

Факультет: Комп'ютерні технології і системи

Кафедра: ЕОМ

Рівень вищої освіти: Другий (магістерський)

Освітня програма: Комп'ютерна інженерія
Спеціальність: 123 Комп'ютерна інженерія
(шифр та назва)

ЗАТВЕРДЖУЮ

Завідувач кафедри ЕОМ

_____ Ігор ЖУКОВИЦЬКИЙ

(підпис) (Ім'я ПРІЗВИЩЕ)

Дата 12.10.2022

ЗАВДАННЯ

на кваліфікаційну роботу магістра
(ступінь вищої освіти)

студенту Нікітіну Вячеславу Миколайовичу
(Прізвище, Ім'я По батькові)

1. Тема роботи: Дослідження та розробка засобів вивчення решіткового кодування.

Керівник роботи: Жуковицький Ігор Володимирович, професор
(Прізвище, Ім'я, По батькові, науковий ступінь, вчене звання)

затверджені наказом від "14" 10 2022 р. № 1082 ст.

2. Строк подання студентом роботи: 20.12.2022 р.

3. Вихідні дані до роботи: Для розробки та дослідження засобів використати Microsoft Visual Studio, наявні методичні засоби з відповідною літературою

4. Зміст пояснювальної записки (перелік питань, які потрібно опрацювати):

4.1 Аналітична частина: Зробити аналіз засобів вивчення решіткового кодування, використання наявних сучасних методів використання треліс-модуляції за методикою згорткового кодування.

4.2 Основна частина: Розробка модифікації методики вивчення trellis-кодування в курсі ТІК. Дослідження алгоритму Вітербі, алгоритму Вітербі м'якого рішення, методу Клода Шеннона, згорткового кодування.

КАЛЕНДАРНИЙ ПЛАН

Назва розділу	Термін виконання	Обсяг розділу, %
Аналіз засад решіткового кодування та його використання в сучасних системах передачі інформації		20%
Аналіз підходів до вивчення решіткового кодування в курсі «Теорія інформації та кодування»		20%
Розробка програмного лабораторного стенда для вивчення		20%
Розробка модифікації методики вивчення trellis-кодування в курсі ТІК		20%
Вступ, Висновки		10%
Оформлення роботи, підготовка демонстраційних матеріалів та доповіді		10%

Дата видачі завдання: «12» 10 2022 р.

Керівник дипломної роботи _____
Жуковицький І.В.
(підпис) (ПІБ)

Завдання прийняв до виконання _____ Нікітін В.М.
(підпис) (ПІБ)

Факультет _Комп'ютерних технологій і систем__ кафедра _____ЕОМ_____

Спеціальність __Комп'ютерна інженерія_____

«ЗАТВЕРДЖУЮ»

Завідувач кафедри

_____

(підпис)

«___»_____20_ р.

ЗАВДАННЯ

до дипломної роботи на здобуття освітнього ступеня _____магістр_____
(освітнього ступеня)

студента групи _КС2021__ Нікітіна Вячеслава Миколайовича_____
(номер групи) (ПІБ)

1 Тема дипломної роботи Дослідження та розробка засобів вивчення решіткового кодування_____
_____
_____
_____

затверджена наказом по університету від «17» січня 2021 р. № 57.

2 Термін подання студентом закінченої роботи__21.12.2022 р._____

3 Вихідні дані до дипломної роботи _____
_____
_____
_____

4 Зміст пояснювальної записки (перелік питань до розробки) _____

Вступ_____
1. Аналіз засад решіткового кодування та його використання в сучасних системах передачі інформації_____
2. Аналіз підходів до вивчення решіткового кодування в курсі «Теорія інформації та кодування»_____
3. Розробка програмного лабораторного стенда для вивчення_____
4. Розробка модифікації методики вивчення trellis-кодування в курсі ТІК_____
Висновки_____

5 Перелік креслень (демонстраційного матеріалу) Комп'ютерна презентація за результатом виконання роботи._____

Зміст

Анотація

У цій роботі ми зосереджуємось виключно на протоколі обчислення та пересилання на основі канального кодування для мережевого кодування фізичного рівня.

Основний принцип цієї стратегії ретрансляції заснований на використанні решітчастого кодування. Вихідні вузли в ретрансляційній мережі кодують свої повідомлення в кодові слова решітки і передають їх ретранслятору. Останній отримує шумне змішування цих кодових слів і декодує цілу лінійну їх комбінацію для послідовної передачі.

Наскільки нам відомо, усі існуючі роботи, пов'язані з протоколом обчислити та переслати, вивчають лише його теоретичні межі, і досі не було запропоновано жодного експериментального аналізу. Наш внесок у цю роботу стосується безлічі практичних аспектів, пов'язаних із решітчастим декодуванням для обчислення та пересилання, які необхідно вирішити, щоб досягти багатообіцяючого потенціалу цієї стратегії.

Ми пропонуємо практичні підходи до кодування та декодування та досліджуємо досягнутий порядок рознесення та визначаємо відповідні параметри, які можуть на нього вплинути. Ми надаємо результати моделювання, щоб порівняти продуктивність різних запропонованих підходів до кодування та декодування та зв'язати теоретичні результати з практичними аспектами.

РЕФЕРАТ

Пояснювальна записка до дипломного проекту: 97 сторінок, 26 малюнків, 15 формул, 23 літературних джерел та 6 додатків.

Об'єктом дослідження є організація вивчення методик решіткового кодування в курсі “Теорія інформації та кодування”.

Предметом дослідження є підходи до методик вивчення решіткового кодування , зокрема з використанням методичних вказівок з курсу “Теорії інформації та кодування”.

Метою кваліфікаційної роботи є реалізація дослідження згорткових кодів на прикладі розробки програмної частини та кодування відправляємих сигналів.

Задачі дослідження**:**

- виявлення принципів модуляції та завадостійкості сигналів, розбір сучасних складних видів модуляції;
- розробка методики поєднання модуляції та кодування, побудування сигнально-кодових конструкцій ;
- аналіз підходів до вивчення решіткового кодування в курсі “Теорія інформації та кодування”;
- рекомендації щодо змін методики вивчення trellis-кодування.

Галуззю застосування даної кваліфікаційної роботи є розробка простого програмного інтерфейсу, який кодує повідомлення.

TRELLIS-КОДУВАННЯ, ПІДХОДИ ДО УДОСКОНАЛЕННЯ ВИВЧЕННЯ РЕШІТКОВОГО (TRELLIS-) КОДУВАННЯ, ПОЄДНАННЯ МОДУЛЯЦІЇ ТА КОДУВАННЯ, ПРОБЛЕМИ ДЕКОДУВАННЯ ЗГОРТКОВИХ КОДІВ, РЕКОМЕНДАЦІЇ ЩОДО ЗМІН ВИВЧЕННЯ TRELLIS-КОДУВАННЯ, СУЧАСНІ НАПРЯМКИ ВИКОРИСТАННЯ, РОЗРОБКА МОДИФІКАЦІЇ ВИВЧЕННЯ TRELLIS-КОДУВАННЯ В КУРСІ “ТІК”.

Вступ

Актуальність роботи:

Решіткове кодування в у теперішньому часі широко використовується у двох аспектах - як елемент згорткового кодування, де для аналізу використовуються решіткові діаграми; як елемент trellis-кодування при складанні сигнально-кодових конструкцій. Студенти нашої спеціальності знайомляться з цими напрямками в курсі “Теорія інформації та кодування”. Досвід навчання показує, що ці теми виявляються досить складними для вивчення та методика їх необхідна бути досконалою.

У налаштуваннях бездротового зв’язку виникає кілька проблем з передачею, яких немає в дротовому корпусі. Перш за все, розподіл ресурсів представляє велику справу, оскільки багато користувачів використовують один і той же канал, а потім стає важче однаково дозволити різним користувачам, які мають різні потреби та обмеження, отримати доступ до бездротового середовища. Крім того, з еволюцією бездротових додатків і мереж попит зростав, але в той же час спектральних ресурсів недостатньо і недостатньо для задоволення всіх запитів. Крім того, порівняно з дротовими мережами, в бездротових каналах енергоефективність є важливим обмеженням, яке слід враховувати, оскільки бездротові термінали використовують зарядні батареї та повинні споживати якомога менше енергії, щоб бути достатньо надійними. Нарешті, у бездротових мережах носій страждає від багатопроменевого згасання, а також від перешкод через властивості трансляції та суперпозиції. Всі ці проблеми призводять до втрати продуктивності.

Об’єкт дослідження:

Об

З цього виникає інтерес до кодування бездротової мережі, заснованого на використанні перешкод, що надаються каналом, для покращення продуктивності мережі.

Предмет дослідження

Кодування бездротової мережі перетворює властивості широкомовної передачі та суперпозиції в характеристики підвищення для досягнення більш

високих швидкостей передачі та збільшення пропускної здатності мережі. Кілька робіт були зацікавлені в мережевому кодуванні у бездротових фреймворках. При застосуванні на фізичному рівні кодування бездротової мережі називається мережевим кодуванням фізичного рівня і може виконуватися спільно з методами модуляції-демодуляції і канальним кодуванням. Перша відмінність між такими методами та кодування бездротової мережі, що виконується на верхньому рівні, полягає в тому, що вихідні вузли надсилають свої повідомлення одночасно, а не послідовно, друга відмінність полягає в тому, що мережевим кодуванням фізичного рівня є мережеве кодування на основі сигнального простору. Фактично, реле змішує отримані сигнали без декодування кожного з них окремо, на відміну від випадку мережевого кодування, де ретранслятор декодує пакети, а потім об'єднує декодовані біти для послідовної передачі.

Мета

У цій роботі ми зосереджуємось виключно на стратегії кодування мережі фізичного рівня на основі кодування каналів під назвою «Обчислювати та переадресовувати» (обчислити та переслати). Цей протокол був першим запропонований Гастпаром і Назером для каналу ретрансляції множинного доступу, де k користувачів передають n багатовимірних інформаційних повідомлень до одного або кількох пунктів призначення за допомогою вузлів ретрансляції. Останні використовують шумну лінійну комбінацію вихідних повідомлень, надану каналом, для декодування безшумних цілочисельних лінійних рівнянь з них і пересилання їх до вузла призначення. Отримавши достатню кількість лінійних рівнянь вихідних переданих повідомлень, пункт призначення декодує потрібні повідомлення. Запропонована схема заснована на вкладених решітчастих кодах, оскільки вони мають хороші статистичні та алгебраїчні властивості структури, що дозволяють досягти пропускної здатності каналів «точка-точка» та продуктивності стандартного випадкового кодування. Цікавою властивістю цих кодів є те, що вони гарантують, що цілі комбінації переданих кодових слів самі є кодовими словами.

Задачі

-розглянути практичну реалізацію протоколу обчислити та переслати для випадку 2 користувачів.

- дослідити випадки одновимірного вихідного сигналу, в якому вузли джерела передають скалярні реальні сузір'я.

- розглянути випадок двовимірного вихідного сигналу з використанням конструкції ST-кодування, запропонованої для каналу множинного доступу (MAC).

Для обох схем ми вивчаємо методи декодування та зосереджуємось на практичних аспектах, які впливають на збільшення різноманітності. Результати моделювання показують, що: по-перше, стратегія ретрансляції обчислити та переслати відповідає своєму обіцяному потенціалу і забезпечує кращу продуктивність, ніж схема декодування MAC. Тоді досягнутий коефіцієнт рознесення залежить від вибору коефіцієнтів декодованої функції на ретрансляторі, а також від розміру сузір'я для одновимірної схеми та від коду решітки для багатовимірного випадку.

Зміст роботи включає реферат, вступ, чотири розділ основної частини, висновки, перелік посилань та два додатки.

У вступі надана загальна характеристика роботи.

В розділі 1 відбувається ознайомлення з решітковим кодуванням

В розділі 2 розглядаються моделі системи та припущення.

В розділі 3 йдеться про протокол обчислити та переслати та різні кроки, необхідні для його виконання. Ми зосереджуємось виключно на підходах до решіткового декодування та досліджуємо параметри, які впливають на їх продуктивність.

В розділі 4 розглядається випадок одновимірних реальних сузір'їв, ми проаналізуємо порядок різноманітності обчислити та переслати і представимо деякі результати моделювання. Також розглядається приклад проектування двовимірного решіткового кодування.

У висновках сформульовані основні результати роботи.

В додатках наведені практична частина реалізації програми та результати розрахунків, що виконувались при решітковому кодуванні.

Наукова новизна ґрунтується на підставі аналізу існуючих рішень та рекомендацій виявлені принципи щодо раціональної організації решіткового кодування та особливості його застосування на практиці.

Практична цінність полягає у тому, що на базі дослідити та покращити методи решіткового кодування.

1. АНАЛІЗ ЗАСАД РЕШІТКОВОГО КОДУВАННЯ ТА ЙОГО ВИКОРИСТАННЯ В СУЧАСНИХ СИСТЕМАХ ПЕРЕДАЧІ ІНФОРМАЦІЇ

1.1. Решіткове кодування як різновид згорткового кодування із захистом від помилок

1.1.1. Загальна характеристика кодів із захистом від помилок

Принципові особливості передачі повідомлень дискретним каналом з помилками пояснює рис. 1.6:

рис. 1.1– канали з помилками

- для канального кодування Клод Шеннон сформулював та довів теорему, аналогічну до тієї, яку ми вивчали для стиснення повідомлень. Суть теореми Шеннона у тому, що у принципі можливо закодувати повідомлення в такий спосіб, щоб смуга пропускання каналу використовувалася максимально повно (у своїй подолання обмеження пропускної спроможності шляхом кодування неможливо);

- образно кажучи, кодування передачі по каналу з помилками створює «захисну оболонку» для інформаційного коду, що передається. «Сила» такої оболонки визначається надмірністю коду, що генерується. У мінімальній версії дозволяє виправити «поломки», а максимальної - ще й усунути їх;

- зазвичай існує два рівні кодування: стиснення повідомлення називається кодуванням джерела, а захист від помилок називається кодуванням каналу. Перший із рівнів усуває вихідну надмірність. З іншого боку, навпаки, додається надмірність коду – але це дуже корисно, необхідно для усунення помилок передачі;

– теорема Шеннона вказує на те, що за рахунок надлишкового кодування можна забезпечити надійну передачу за рахунок збільшення її тривалості. Зменшення пропускної здатності каналу означає збільшення необхідного часу передачі (так само, як заданий об'єм рідини може пройти через менший поперечний переріз труби за більш тривалий час).

Найпростіший спосіб перекодувати – використовувати один контрольний біт. Якщо під час передачі виникає помилка в одній із цифр, правило парності порушується і, таким чином, виявляється помилка. Звичайно, цей метод не дозволяє визначити положення помилки, а це означає, що код не може виправити виявлену помилку. Більше того, якщо під час передачі "впливають" одночасно на два біти коду, умова парності не порушується, і така помилка не буде виявлена[34];

Ви можете збільшити можливості коду (його коригуючу здатність), якщо використовуєте більше контрольних бітів. Застосування трьох перевірок одночасно дозволяє визначити положення хибного розряду, отже, виправити помилку, інвертуючи її. Такі коди (вони називають кодами Хеммінга) використовуються для усунення результатів збоїв пам'яті сервера;

- при однаковій кількості керуючих бітів відповідна коригувальна сила коду може бути спрямована не на виправлення окремих помилок (які характерні, зокрема, для ОЗУ комп'ютера), а на виявлення їх ланцюжків, які часто зустрічаються при передачі через лінії зв'язку).

Використання додаткових бітів у принципі дозволяє виявляти і навіть виправляти помилки передачі. У цьому випадку збільшення коригуючої здатності коду вимагає збільшення надмірності. Однак на практиці ймовірність помилок може значно змінюватись (наприклад, при широкомовній передачі), і

очевидно, що створювати коди, розраховані на найгірший випадок, недоцільно. Це протиріччя вирішує інший підхід усунення помилок передачі [34].

1.1.2 Згорткові коди

Наведемо короткий опис згортокового решіткового кодування зокрема за матеріалами [2,4].

Згортковий код вводить зайві біти в потік даних за допомогою регістрів лінійного зсуву, як показано на рис. 1.1.

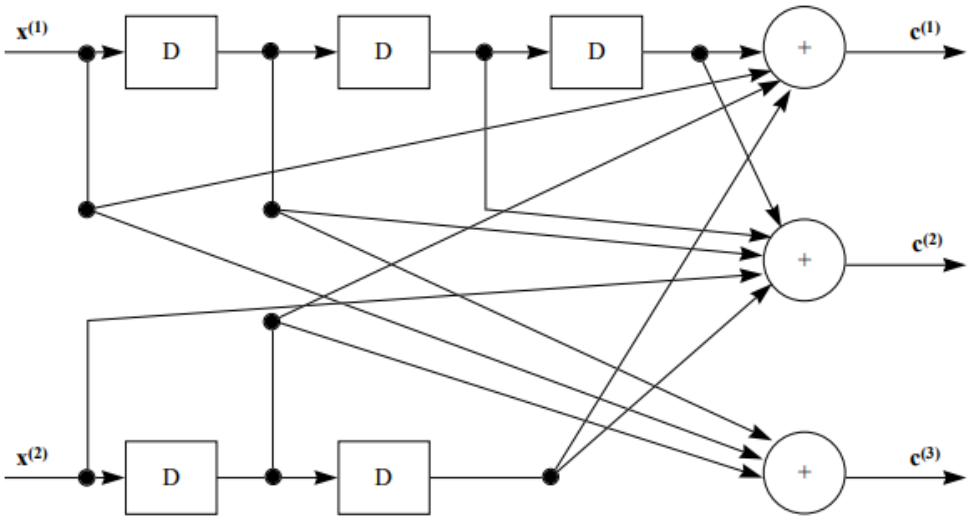

Рис. 1.2. - Приклад згорткового кодера, де $x^{(i)}$ – це вхідний інформаційний бітовий потік, а $c^{(i)}$ – вихідний закодований бітовий потік

Інформаційні біти вводяться в регістри зсуву, а вихідні закодовані біти отримують шляхом додавання по модулю 2 вхідних інформаційних бітів і вмісту регістрів зсуву.

Швидкість коду r для згорткового коду визначається як

$$r = \frac{k}{n}$$

де k - кількість паралельних вхідних інформаційних бітів, а n - кількість паралельних вихідних закодованих бітів за один часовий інтервал. Довжина обмеження K для згорткового коду визначається як

$$K = m + 1$$

де m - максимальна кількість етапів (розмір пам'яті) у будь-якому регістрі

зсуву. Регістри зсуву зберігають інформацію про стан згорткового кодера, а довжина обмеження пов'язує кількість бітів, від яких залежить вихід. Для згорткового кодера, показаного на рис. 1.1, швидкість кодування r=2/3, максимальний розмір пам’яті m=3 і довжина обмеження K=4.

Для опису властивостей коду буде використаний простий згортковий код, як показано на рисунку 1.2.

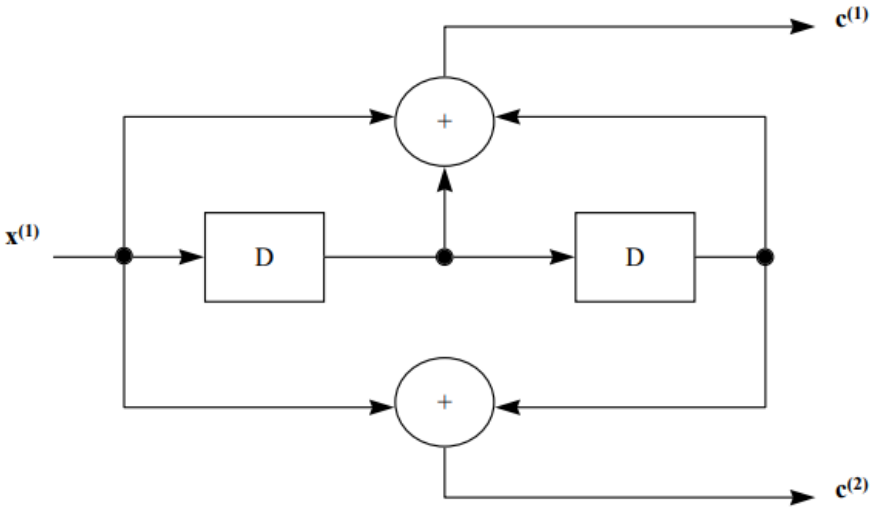

Рисунок 1.3. - Згортковий кодер з k=1, n=2, r=1/2, m=2 та K=3

Кодер може бути представлений кількома різними, але еквівалентними способами:

1. Представлення генератора
2. Дерево діаграми
3. Діаграма стану
4. Решітчаста діаграма

Представлення генератора

Представництво генератора показує апаратне підключення відводів сдвигового регістра до суматорів по модулю-2. Вектор генератора представляє положення відводів для виходу. «1» означає з'єднання, а «0» означає відсутність з'єднання. Наприклад, два генераторних вектора для кодера на рис. 1.2 є g1 = [111] і g2 = [101], де індекси 1 і 2 позначають відповідні вихідні термінали.

Деревоподібна діаграма показує всю можливу інформацію та закодовані

послідовності для згорткового кодера. На рисунку 1.4 показано дерево-діаграму для кодера на рис. 1.2 для чотирьох інтервалів вхідних бітів

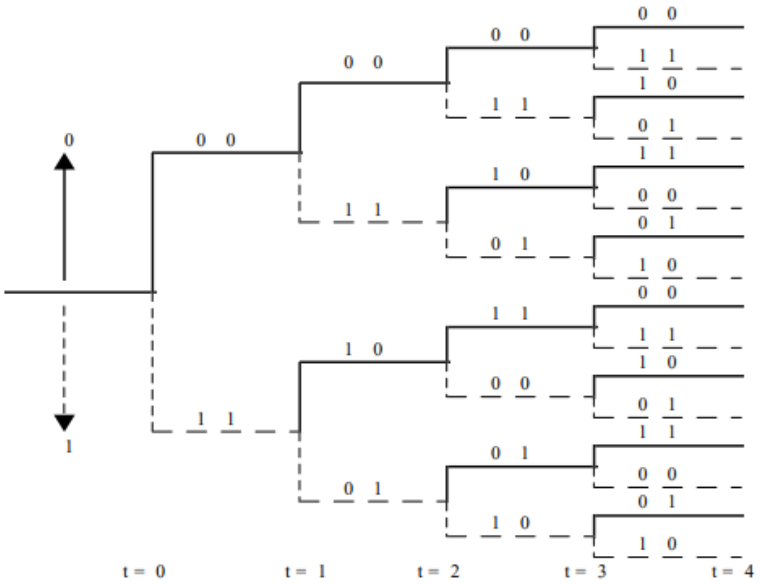

Рис. 1.4. - Дерево-схема представлення кодера на рис. 1.2. для чотирьох інтервалів вхідних бітів

На діаграмі дерева суцільна лінія представляє вхідний інформаційний біт 0, а пунктирна — біт вхідної інформації 1. Відповідні вихідні закодовані біти показані на гілках дерева. Вхідна інформаційна послідовність визначає певний шлях через діаграму дерева зліва направо. Наприклад, послідовність вхідної інформації x={1011} створює вихідну закодовану послідовність c={11, 10, 00, 01}. Кожен біт вхідної інформації відповідає розгалуженню вгору (для біта вхідної інформації 0) або вниз (для біта вхідної інформації 1) у вузлі дерева.

Діаграма стану показує інформацію про стан згорткового кодера. Інформація про стан згорткового кодера зберігається в регістрах зсуву. На рис. 1.4 показана діаграма стану кодера на рис. 1.2.

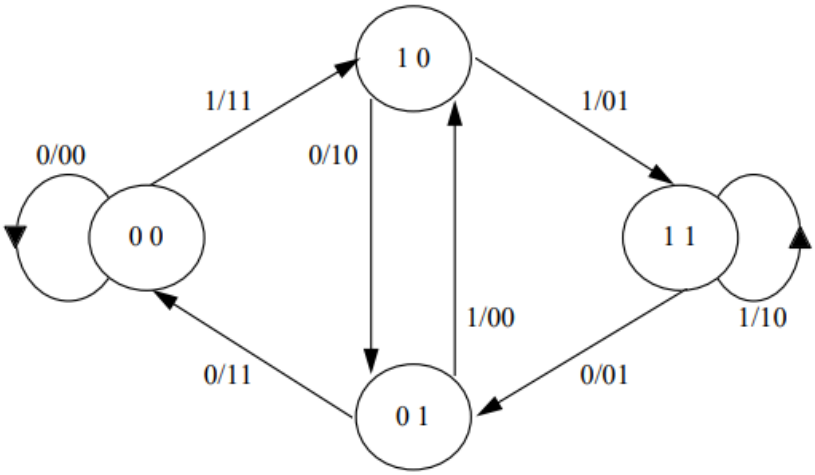

Рис. 1.5 - Подання діаграми стану кодера на рис. 1.2.

На діаграмі стану інформація про стан кодера показана кружечками. Кожен новий біт вхідної інформації викликає перехід з одного стану в інший. Інформація про шлях між станами, позначена як $\frac{x}{c}$, представляє вхідний інформаційний біт x і вихідні закодовані біти c. Зазвичай починати згорткове кодування з нульового стану. Наприклад, вхідна інформаційна послідовність x={1011} (починаючи з нульового стану) призводить до послідовності переходу стану s={10, 01, 10, 11} і створює вихідну закодовану послідовність c={11, 10, 00, 01}. На рис. 1.5. показано шлях, пройдений через стан діаграми для наведеного прикладу.

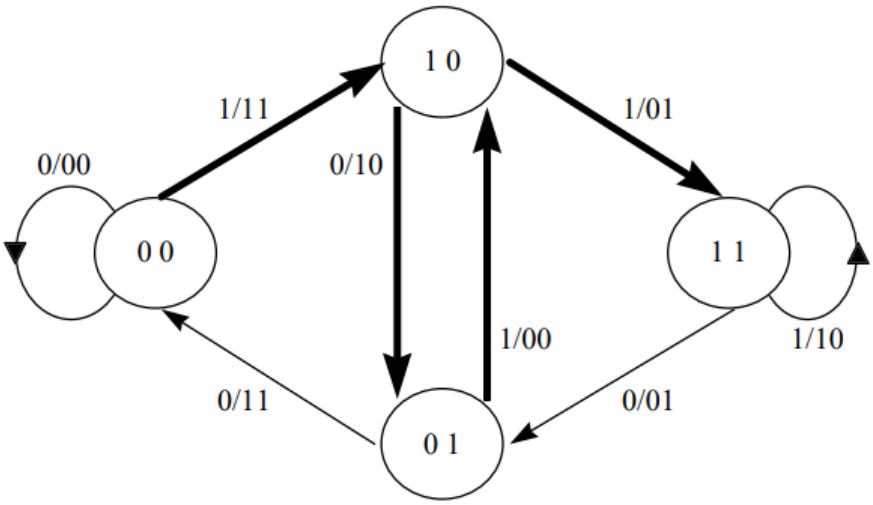

Рис. 1.6. - Переходи стану (шлях) для послідовності вхідної інформації {1011}.

1.1.2. Проблеми декодування згорткових кодів. Алгоритм Вітербі.

Проблема декодування згорткових кодів, є одним з найважливіших напрямків сучасних наукових досліджень, оскільки від її вирішення, за інших рівних умов, залежить достовірність передачі інформації, що особливо актуальною в системах передачі інформації без використання зворотного каналу. ставить питання про розробку методик порівняльної оцінки їх завадостійкості, що максимально повно відображають коригуючу здатність системи.

Алгоритм Вітербі жорсткого рішення

Для згорткового коду вхідна послідовність x «згортається» до закодованої послідовності c. Послідовність c передається через шумовий канал, і отримується послідовність r, що приймається. Алгоритм Вітербі обчислює оцінку максимальної правдоподібності для оціненої кодової послідовності y з отриманої послідовності r таким чином, що максимізує ймовірність p(r|y), що послідовність r буде отримана з урахуванням оціненої кодової послідовності y. Послідовність y має бути однією з допустимих кодових послідовностей і не може бути довільною послідовністю. На рис. 1.8 показана описана структура системи.

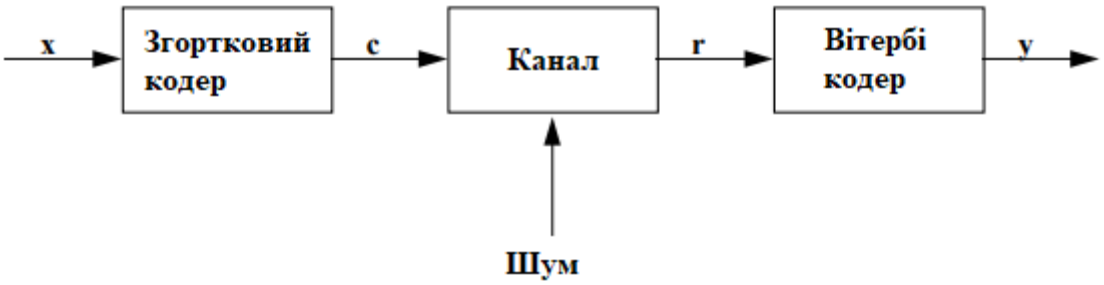

Рис. 1.7 - Система згорткового коду.

Для згорткового коду зі швидкістю r кодер вводить k біт паралельно і виводить n біт паралельно на кожному кроці часу. Вхідна послідовність позначається як

$$\mathbf{x}=(x_0^{(1)}, x_0^{(2)}, ..., x_0^{(k)}, x_1^{(1)}, ..., x_1^{(k)}, x_{L+m-1}^{(1)}, ..., x_{L+m-1}^{(k)})$$

та кодована послідовність позначається як

$$\mathbf{c}=(c_0^{(1)}, c_0^{(2)}, ..., c_0^{(n)}, c_1^{(1)}, ..., c_1^{(n)}, c_{L+m-1}^{(1)}, ..., c_{L+m-1}^{(n)})$$

де L позначає довжину вхідної інформаційної послідовності, а m позначає

максимальну довжину регістрів зсуву. Додаткові m нульових бітів необхідні в хвості інформаційної послідовності, щоб повернути згортковий кодер у стан, що повністю нуль. Необхідно, щоб кодер починався і закінчувався в нульовому стані. Нижній індекс позначає індекс часу, а верхній індекс позначає біт у певному вхідному k-бітовому або вихідному n-бітовому блоці. Отримані та оцінені послідовності r і y можна описати аналогічно

$$\mathbf{r}=(r_0^{(1)}, r_0^{(2)}, ..., r_0^{(n)}, r_1^{(1)}, ..., r_1^{(n)}, r_{L+m-1}^{(1)}, ..., r_{L+m-1}^{(n)})$$

та

$$\mathbf{y}=(y_0^{(1)}, y_0^{(2)}, ..., y_0^{(n)}, y_1^{(1)}, ..., y_1^{(n)}, y_{L+m-1}^{(1)}, ..., y_{L+m-1}^{(n)}).$$

Для декодування максимальної ймовірності алгоритм Вітербі вибирає y, щоб максимізувати p(r|y). Вважається, що канал не має пам'яті, і, таким чином, шумовий процес, що впливає на прийнятий біт, не залежить від процесу шуму, що впливає на всі інші отримані біти. З теорії ймовірностей ймовірність спільних незалежних подій еквівалентна добутку ймовірностей окремих подій. Таким чином,

$$p(\mathbf{r}|\mathbf{y}) = \prod_{i=0}^{L+m-1}[p(r_i^{(1)}|y_i^{(1)})p(r_i^{(2)}|y_i^{(2)})\cdots p(r_i^{(n)}|y_i^{(n)})] = \prod_{i=0}^{L+m-1}\left(\prod_{j=1}^{n} p(r_i^{(j)}|y_i^{(j)})\right)$$

Це рівняння називається функцією правдоподібності y, якщо отримано r. Оцінка, яка максимізує p(r|y), також максимізує log p(r|y), оскільки логарифми є монотонно зростаючими функціями. Таким чином, логарифмічна функція ймовірності може бути визначена як

$$\log p(\mathbf{r}|\mathbf{y}) = \sum_{i=0}^{L+m-1}\left(\sum_{j=1}^{n} \log p(r_i^{(j)}|y_i^{(j)})\right)$$

Для спрощення маніпулювання підсумовуванням за функцією журналу визначено бітову метрику. Розрядна метрика визначається як

$$M(r_i^{(j)}|y_i^{(j)}) = a[\log p(r_i^{(j)}|y_i^{(j)}) + b]$$

де a і b вибираються таким чином, що бітова метрика є невеликим натуральним числом. Значення a і b визначені для бінарно-симетричного каналу або декодування з жорстким рішенням. На рис. 1.9 показано бінарно-симетричний канал.

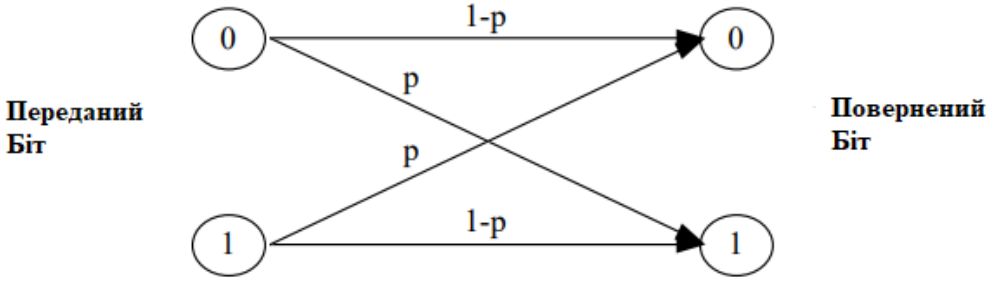

Рис. 1.8 - Бінарна симетрична модель каналу, де p – ймовірність кросовера

Для цього випадку алгоритм Вітербі вибирає кодову послідовність y через решітку, яка має найбільшу відстань вартість/Хеммінга щодо отриманої послідовності r. Крім того, для довільного каналу значення a і b знаходять на основі пробних помилок, щоб отримати прийнятну бітову метрику.

З бітової метрики визначається метрика шляху. Показник шляху визначається як

$$M(\mathbf{r}|\mathbf{y}) = \sum_{i=0}^{L+m-1}\left(\sum_{j=1}^{n} M(r_i^{(j)}|y_i^{(j)})\right)$$

і вказує загальну вартість оцінки прийнятої бітової послідовності r з декодованою бітовою послідовністю y на решітковій діаграмі. Крім того, метрика k-ї гілки визначається як

$$M(\mathbf{r}_k|\mathbf{y}_k) = \sum_{j=1}^{n} M(r_k^{(j)}|y_k^{(j)})$$

і k-та метрика часткового шляху визначається як

$$M^k(\mathbf{r}|\mathbf{y}) = \sum_{i=0}^{k} M(\mathbf{r}_i|\mathbf{y}_i) = \sum_{i=0}^{k}\left(\sum_{j=1}^{n} M(r_i^{(j)}|y_i^{(j)})\right)$$

Показник k-ї гілки вказує вартість вибору гілки за діаграмою решітки. K-та метрика часткового шляху вказує вартість вибору частково декодованої бітової послідовності y до індексу часу k.

Алгоритм Вітербі використовує решітчасту діаграму для обчислення показників шляху. Кожному стану (вузлу) на решітковій діаграмі присвоюється значення, метрика

часткового шляху. Показник часткового шляху визначається із стану s = 0 у момент t = 0 до конкретного стану s = k в момент t ≥ 0. У кожному стані вибирається «найкраща» метрика часткового шляху з шляхів, закінчених у цьому стані. «Найкращою» метрикою часткового шляху може бути як більша, так і менша метрика, залежно від того, вибрані a та b традиційно чи альтернативно.

Вибрана метрика представляє шлях уцілілих, а решта показників – шляхи, які не вижили. Шляхи, які вижили, зберігаються, а шляхи, які не вижили, відкидаються на решітковій діаграмі. Алгоритм Вітербі вибирає єдиний шлях виживання, що залишився в кінці процесу, як шлях максимальної ймовірності. Відстеження шляху максимальної ймовірності на решітковій діаграмі тоді забезпечить декодовану послідовність максимальної ймовірності.

Складність декодування для згорткових кодів

Для загального згорткового коду вхідна інформаційна послідовність містить k*L бітів, де k — кількість паралельних інформаційних бітів за один часовий інтервал, а L — кількість інтервалів часу. Це призводить до L+m етапів на шпалерній діаграмі. У решітковій діаграмі є рівно 2k*L різних шляхів, і в результаті вичерпний пошук послідовності ML матиме обчислювальну складність порядку O[2k*L]. Алгоритм Вітербі зменшує цю складність, виконуючи пошук ML по одному етапу в решітці. У кожному вузлі (стані) шпалери є 2k обчислень. Кількість вузлів на етапі в шпалері 2м. Отже, складність алгоритму Вітербі має порядок O[(2k)(2m)(L+m)]. Це значно зменшує кількість обчислень, необхідних для реалізації декодування максимальної ймовірністі, оскільки кількість часових інтервалів L тепер є лінійним коефіцієнтом, а не коефіцієнтом ступеня складності. Однак складність буде експоненціально зростати, якщо збільшується k або m.

Алгоритм Вітербі м'якого рішення

Існує два загальні методи реалізації алгоритму Вітербі з м’яким рішенням. Перший метод (метод 1) використовує метрику евклідової відстані замість метрики відстані Хеммінга. Отримані біти, які використовуються в метриці евклідової відстані, обробляються за допомогою багатобітового квантування. Другий метод (Метод 2) використовує кореляційну метрику, де отримані біти, що використовуються в цій метриці, також обробляються за допомогою багатобітового квантування.

Алгоритм Вітербі м'якого рішення

При декодуванні з м’яким рішенням приймач не призначає нуль або одиницю (однобітове квантування) кожному отриманому біту, а використовує багатобітові або нескінченно-бітові квантовані значення. В ідеалі отримана послідовність r є нескінченно-бітовим квантуванням і використовується безпосередньо в декодері Вітербі з м’яким рішенням. Алгоритм Вітербі з м'яким рішенням подібний до його алгоритму жорсткого рішення, за винятком того, що в метриці замість відстані Хеммінга використовується квадрат Евклідової відстані.

Алгоритм Вітербі м’якого рішення може бути реалізований наступним чином:

$S_{k,t}$ - це стан на решітчастій діаграмі, який відповідає стану S_k в момент часу t. Кожному стану в решітці присвоюється значення, що позначається $V(S_{k,t})$.

1. (a) Час ініціалізації $t = 0$.

(b) Ініціалізуйте $V(S_{0,0}) = 0$ і всі інші $V(S_{k,t}) = +\infty$.

2. (a) Установіть час $t = t+1$.

(b) Обчисліть метрику часткового шляху для всіх шляхів, що переходять у стан S_k у момент часу t. Спочатку знайдіть метрику t-ї гілки

$$M(\mathbf{r}_t|\mathbf{y}_t) = \sum_{j=1}^{n} M(r_t^{(j)}|y_t^{(j)})$$

1. Це обчислюється з квадрата евклідової відстані

$$\sum_{j=1}^{n}(r_t^{(j)} - y_t^{(j)})^2$$

По-друге, обчисліть метрику часткового шляху t

$$M^t(\mathbf{r}|\mathbf{y}) = \sum_{i=0}^{t} M(\mathbf{r}_i|\mathbf{y}_i)$$

Це розраховується з

$$V(S_{k,t-1}) + M(\mathbf{r}_t|\mathbf{y}_t)$$

3. (a) Встановіть $V(S_{k,t})$ як «найкращу» метрику часткового шляху, яка переходить у стан S_k у момент t. Умовно «найкращим» показником часткового шляху є показник часткового шляху з найменшим значенням.

(b) Якщо є рівність для «найкращої» метрики часткового шляху, тоді може бути обрана будь-яка з пов’язаних метрик часткового шляху.

4. Збережіть «найкращу» метрику часткового шляху та пов'язані з нею біт і шляхи стану, що вижив.

5. Якщо $t < L+m-1$, поверніться до кроку 2.

На практиці використовуються різні методи для обробки декодування Вітербі на комп'ютері. Вони включають нормалізацію параметрів завдяки фіксованому діапазону в межах комп’ютера, використовуючи поріг як основу, щоб потім відняти значення T з кожної метрики. Існує також спосіб збереження пам'яті шляху, щоб мати можливість витягувати інформаційні біти. Для відновлення послідовності станів у зворотному порядку зберігається пам’ять відстеження, що використовує значення рішень, які вказують на переходи станів. Це використовується, коли код реалізований апаратно.

1.1.3. Сучасні напрямки використання

При розробці сучасних систем передачі інформації однією з найважливіших проблем є проблема вибору типу і параметрів використовуваного коду. іншого алгоритму декодування ґрунтується на порівнянні певного підмножини існуючих алгоритмів, з урахуванням встановлених обмежень

Для побудови основної оцінки системи передачі інформації - залежності ймовірності невиявленої помилки декодування від співвідношення сигнал шум, в системі зв'язку використовує згорткові коди, застосовуються аналітичні та статистичні методи.

Аналітичні методи. Засновані на побудові оцінок меж ймовірностей помилок кожної конкретної системи передачі

Інформації, т е системи зв'язку, для якої визначено тип і параметри згорткового коду, метод його декодування, а також тип використовуваного каналу зв'язку

Слід особливо відзначити, що практично застосовні вирази, однозначно визначають ймовірність помилки на символ для всього ансамблю згорткових кодів до цього часу не отримані. були отримані методами комп'ютерного моделювання

Відсутність єдиних аналітичних виразів для аналізу згорткових кодів, велика різноманітність алгоритмів декодування, більшість з яких носить імовірнісний характер, визначає широке поширення статистичних методів дослідження згорткових кодів

Статистичні методи (методи статистичних випробувань). Засновані на використанні положень теорії математичної статистики та пов'язані з побудовою імітаційної моделі системи зв'язку, побудованої на заданому згортковому коді, та визначенні основних імовірнісних характеристик системи

Як правило, досить висока точність оцінки, при застосуванні даного методу, може бути отримана лише з умовою проведення великої кількості випробувань, отже, метод доцільно реалізовувати на швидкодіючих ЕОМ Для підвищення достовірності отриманих результатів необхідно використовувати статистичні

вибірки значного обсягу Очевидно, що дослідження систем зв'язку , на основі згорткових кодів, статистичними методами є можливим, а за відсутності єдиних аналітичних підходів для всього різноманіття систем зв'язку, є єдиним методом, що широко застосовується на практиці

Сучасні системи зв'язку повинні мати значний запас перешкодостійкості, що визначає їх дослідження при ймовірності

Крім цього для отримання адекватних результатів, обліку помилок високої кратності, особливо при високих відносинах сигнал шум у каналі зв'язку, необхідно працювати з послідовностями значної довжини. Тому процес моделювання, для накопичення необхідної статистики, навіть на високопродуктивних обчислювальних машинах. займає велику кількість часу Більше того, використання описаного підходу, унеможливлює однозначне повторення статистичного експерименту для іншого коду, іншої системи зв'язку і т.п., а отже, з'являється ймовірність дослідження не всіх можливих станів системи. Стає очевидною необхідність створення імітаційної моделі системи передачі інформації, вільної від зазначених недоліків

Також одним із сучасних напрямків є мережеве кодування.

Мережеве кодування можна розглядати як нову техніку виконання маршрутизації, що дозволяє досягти більш високої пропускної здатності. У випадку лінійного мережевого кодування вихідні пакети є лінійними комбінаціями прийнятих пакетів, прийнятих на кінцевих полях. Проілюструємо цю схему на рис. 1.10

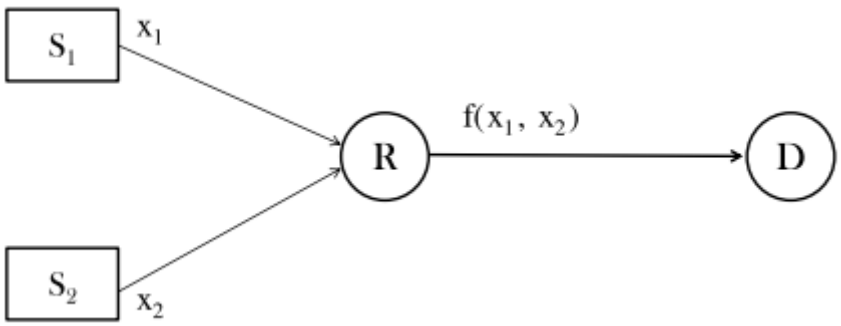

Рис. 1.9. Принцип мережевого кодування.

Синхронізація

Якщо n-кортежи випадають із синхронізації, решітка починає давати безперервні помилкові результати. Це можна перевірити, використовуючи очікувану статистику BER та зростання показників шляху. Цей монітор є зовнішнім щодо декодера. Це досягається за допомогою етапу синхронізації, функцією якого є просування опорної послідовності [v] в декодері, пропускаючи отримані символи (максимум n-1) по одному, поки змінні синхронізації не вказують на нормальну поведінку. Як альтернатива, дані розбиваються на фіксовану довжину (наприклад, кілька тисяч бітів). Потім для синхронізації приймача додається відоме унікальне слово і змушує згортковий кодер повернутися до відомого стану.

Пробиті згорткові коди

Перфорація — це процес систематичного видалення деяких згенерованих кодером бітів. Оскільки решітка однакова, кількість інформаційних бітів однакова. Однак цей прокол призводить до більш високої швидкості кодування, ніж оригінал. Основою проколу є матриця, яка називається матрицею проколу, яка визначає операцію.

1.2. Решіткове (trellis-) кодування при побудуванні сучасних сигнально-кодових конструкцій

Треліс-модуляція (ТММ - Trellis Coded Modulation) являє собою спосіб, який дозволяє забезпечити підвищити швидкість передачі повідомлення зі збереженням рівня стійкості до перешкод. Цей спосіб відрізняється гем, що завадостійке кодування та тип модуляції використовуються спільно. Вибрана відповідним чином пара завадостійкий код - спосіб модуляції часто також називається сигнально-кодова конструкція (СКК).

У цій роботі використано метод згорткового кодування, що використовується для передачі даних за допомогою радіорелейних систем зв'язку, режим треліс-модуляції. Такий спосіб підвищує швидкість передачі в два рази при збереженні рівня стійкості до перешкод.

У цифрових системах зв'язку зазвичай підвищують швидкість передачі шляхом зменшення енергетичної ємності біта, тобто. кількості енергії сигналу, що припадає на один біт інформації. Але що менше енергія, то більша ймовірність того, що біт буде спотворений у каналі під час передачі. Тому при підвищенні швидкості передачі розробники завжди стикаються зі зниженням рівня стійкості до перешкод.

Для підвищення стійкості до перешкод каналу передачі даних у цифрових системах застосовуються коди, що виправляють помилки. Проте дія таких кодів який завжди ефективно, оскільки знижується швидкість передачі.

Коригувальні коди

Поряд із багатопозиційними сигналами для підвищення ефективності системи електричного зв'язку (СЕС) широко використовуються завадові коди. Застосування коригувальних кодів дозволяє підвищити вірність передачі повідомлень або за заданої вірності підвищити енергетичну ефективність системи. Це особливо важливо для систем із малою енергетикою, наприклад, систем супутникового зв'язку.

Насправді використовуються як блокові, і безперервні коди.

Енергетичний виграш від застосування завадостійкого кодування тим більше, чим вище необхідна вірність передачі. Побудова таких високоефективних систем з урахуванням сигнально-кодових конструкцій веде до неминучого збільшення складності системи. Чи не пропускна здатність, а складність є обмежуючим фактором при побудові високоефективних систем. Основна проблема полягає в тому, щоб побудувати систему, яка задовольняє високі показники ефективності, при допустимій складності.

1.2.1. Модуляція та завадостійкість сигналів. Сучасні складні види модуляції

Фундаментальним для всіх бездротових комунікацій є модуляція, процес відображення даних, що підлягають передачі на радіоносій. Більшість бездротових передач сьогодні є цифровими, і з обмеженим доступним спектром тип модуляції є більш важливим, ніж будь-коли.

Головна мета модуляції сьогодні — стиснути якомога більше даних у найменший можливий обсяг спектру. Ця ціль, відома як спектральна ефективність, вимірює, наскільки швидко дані можуть передаватися у призначеній пропускній здатності. Одиницею виміру є біт на секунду на Гц (б/с/Гц). Для досягнення та покращення спектральної ефективності з'явилося безліч методів.

Амплітудний зсув і частотний зсув.

Існує три основних способи модуляції синусоїдальної радіонесучої хвилі: зміна амплітуди, частоти або фази. Більш складні методи поєднують два або більше з цих варіантів для покращення спектральної ефективності. Ці основні форми модуляції досі використовуються з цифровими сигналами.

Основний послідовний цифровий сигнал з двійкових нулів і одиниць, що підлягають передачі, а також відповідні сигнали AM і FM, отримані в результаті модуляції. Існує два типи AM-сигналів: маніпуляція включення-вимкнення і амплітудна маніпуляція.

AM створює бічні смуги вище і нижче несучої, що дорівнює найвищому частотному вмісту модулюючого сигналу. Необхідна смуга пропускання вдвічі перевищує вміст найвищої частоти, включаючи будь-які гармоніки для двійкових імпульсно-модулюючих сигналів.

Частотний зсув переміщує несучу між двома різними частотами, які називаються частотами позначки і простору, або f_m і f_s. FM створює декілька частот бічної смуги вище і нижче несучої. Отримана смуга пропускання є функцією найвищої частоти модуляції, включаючи гармоніки та індекс модуляції, який становить:

$m = \Delta f(T)$

Δf - це відхилення або зсув частоти між частотами позначки та проміжними частотами, або:

$\Delta f = f_s - f_m$

T – бітовий інтервал часу даних або зворотна швидкість передачі даних (1/біт/с).

Менші значення m створюють менше бічних смуг. Популярна версія FSK під назвою мінімальна зсувна клавіатура визначає m = 0,5. Також використовуються менші значення, наприклад m = 0,3.

При фазовій і частотній модуляції спектр сигналу виходить більш складним, чим при амплітудній модуляції, тому що бічних гармонік тут утвориться більш двох, але вони також симетрично розташовані щодо основної несучої частоти, а їх амплітуди швидко зменшуються. Тому ці види модуляції також добре підходять для передачі даних по каналу тональної частоти.

Для підвищення швидкості передачі даних використовують комбіновані методи модуляції. Найбільш розповсюдженими є методи квадратурної амплітудної модуляції (Quadrature Amplitude Modulation, QAM). Ці методи основані на поєднанні фазової модуляції з 8 значеннями величин зміщення фази й амплітудної модуляції з 4 рівнями амплітуди. Однак з можливих 32 комбінацій сигналу використовуються далеко не всі. Наприклад, у кодах Трелліса припустимі всього 6, 7 чи 8 комбінацій для подання вихідних даних, а інші комбінації є забороненими. Така надмірність кодування потрібна для розпізнавання модемом помилкових сигналів, що є наслідком перекручувань через перешкоди на телефонних каналах, які комутуються дуже значними за амплітудою і тривалими за часом.

1.2.2. Поєднання модуляції та кодування. Побудування сигнально-кодових конструкцій

В якості завадостійких кодів можуть застосовуватися як блокові, так і згорточні коди. Найбільша ефективність систем передачі з СКК досягається при використанні згорточних кодів у сполученні з ансамблями АФМ-сигналів. Їх енергетичний виграш у гауссівському каналі становить 3...6 дБ залежно від складності об'єднаної системи модуляції і кодування.

Аналіз ефективності систем радіозв'язку показує, що використання багатопозиційної модуляції хоча і дозволяє підвищити швидкість передачі в порівнянні з двійковою модуляцією, але не дозволяє наблизитися до пропускної здатності ні зачастотною, ні за енергетичною ефективністю.

Використання ж коригувальних кодів разом із двійковою модуляцією дозволяє наблизитися до пропускної здатності близької до границі Шеннона.

У реальних багатопроменевих каналах зв'язку крім адитивного шуму виникає міжсимвольна інтерференція (МСІ), викликана пам'яттю каналів. Реакція каналу на послідовність вхідних сигналів викликає взаємне накладення сигналів на виході каналу. Якщо нормувати за потужністю амплітудно-частотну характеристику каналу, то можна сказати, що МСІ призводить до значної зміни відстаней між сигналами на виході каналу і, що особливо важливо, до зменшення мінімальної відстані між ними.

При синтезі сигналів і кодів для каналів із МСІ цей ефект, як правило, не враховується, тобто в якості вхідних сигналів вибираються такі, які погоджені з ідеальним каналом без МСІ. Однак МСІ прагнуть враховувати при синтезі оптимального приймача (декодера). Широко відомим рішенням такого роду є алгоритм Вітербі і його модифікація, яка враховує згорточне кодування .

Розглянемо підхід до кодування в каналах із МСІ, заснований на синтезі таких сигнально-кодових конструкцій, які враховують «деформацію» простору сигналів при передачі по реальному каналу.

Основою цього підходу є можливість перетворення каналів із МСІ в сукупність гауссовських каналів без пам'яті, тобто без МСІ, але відрізняющихся один від одного

скалярним коефіцієнтом передачі або відношенням сигнал/шум.

Сутність методики полягає в побудові сигнально-кодової конструкції з оптимальними за критерієм максимуму частотної ефективності параметрами при обмеженні на значення ймовірності помилкового приймання сигналів в умовах селективних завмирань і нелінійних спотворень.

1.2.3. Сучасні напрямки використання

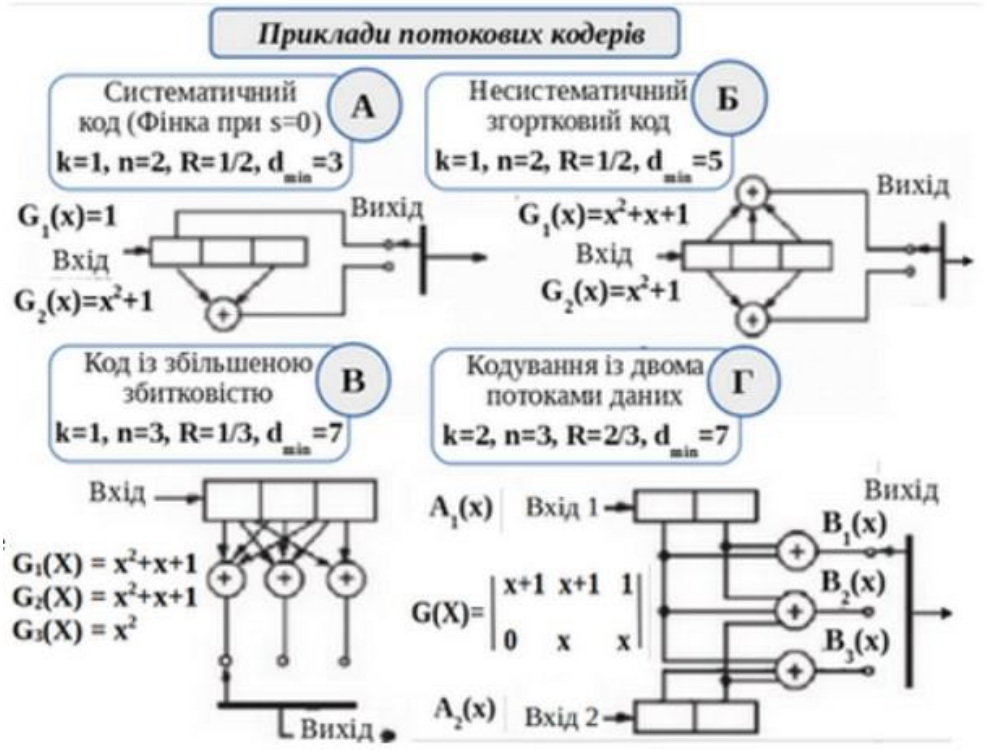

Рис. 1.10. Сучасні напрямки використання решіткового кодування

Прагнення до більшої спектральної ефективності.

Оскільки спектр є кінцевою сутністю, його завжди бракує. Федеральна комісія зі зв'язку та інші державні органи протягом багатьох років призначали більшу частину спектру електромагнітних частот, і більша частина цього активно використовується.

Зараз у секторах стільникового та наземного мобільного радіозв’язку існує дефіцит, що гальмує розширення таких послуг, як висока швидкість передачі даних, а також додавання нових абонентів. Одним з підходів до вирішення проблеми є підвищення ефективності використання, залучаючи більше користувачів у той самий або менший спектр і досягаючи більш високої швидкості передачі даних. Покращені методи модуляції та доступу можуть допомогти.

Однією з найбільш переповнених зон спектру є наземний мобільний радіо і

приватний мобільний радіо, який використовується федеральним урядом, урядами штатів та місцевими органами громадської безпеки, такими як пожежні та поліцейські служби.

Нове обладнання може використовувати аналогову або цифрову модуляцію. Можна помістити стандартний аналоговий FM в канал 12,5 кГц шляхом регулювання індексу модуляції та використання інших методів звуження смуги пропускання. Однак аналоговий FM на каналі 6,25 кГц непрацездатний, тому необхідно використовувати цифрову техніку.

Цифрові методи оцифровують голосовий сигнал і використовують методи стиснення, щоб створити послідовний цифровий сигнал з дуже низькою швидкістю, який можна модулювати у вузьку смугу. Очікується, що такі методи цифрової модуляції відповідатимуть цілі вузькосмугового зв'язку та забезпечать деякі додаткові переваги продуктивності.

2. Аналіз підходів до вивчення решіткового кодування в курсі «Теорія інформації та кодування»

Решіткові коди дозволяють досягти пропускної здатності каналу AWGN; вони використовують ту саму реальну алгебру, що й канал AWGN; і вони мають алгебраїчну структуру, що робить їх придатними для мережевого кодування фізичного рівня, обчислення і пересилання, вирівнювання перешкод тощо [3]–[7]. Ці інформаційно-теоретичні результати ґрунтуються на випадкових конструкціях високорозмірних пар «хороших» решіток: одна решітка забезпечує виграш кодування для каналу AWGN, а інша решітка забезпечує підсилення формування.

Останні роки ознаменувались розробкою практичних, малоскладних ґратчастих кодів із хорошим посиленням кодування [8]–[12]. Однак для того, щоб ці решітки з високим коефіцієнтом посилення кодування можна було застосувати на практиці, вони повинні задовольняти обмеження потужності. В області решітки обмеження потужності задовольняється шляхом вибору набору точок решітки кодування, які знаходяться в конкретній області формування. Одним з підходів для вирішення цього завдання є використання дискретного гаусса n-формування, як зазначено в [2], [13]. Виходячи з концепції [2], у [14] використано дискретне гауссове формування. Інший підхід, названий системним формуванням, був запропонований у [15]. Хоча систематичне формування само по собі не забезпечує виграшів у формуванні, в [15] було запропоновано використовувати систематичне формування разом із решіткою [16] або відображенням оболонки [17], щоб отримати виграш у формі. Інший спосіб виконання формоутворення - це формування вкладеної решітки, в якому область Вороного високорозмірної підрешітки е використовується як область формування. Однією з переваг формування вкладеної решітки перед іншими методами формування є збереження алгебраїчної структури між повідомленнями та кодами решітки [3]. Однак використання великогабаритної решітки для формування є дорогим, оскільки складність формування різко зростає із

збільшенням розміру. Крім того, розробка великогабаритних решіток з гарним збільшенням форми є складним завданням.

Модель системи

Ми розглядаємо модель каналу AWGN. Джерело кодує вхідну інформацію $b \in Z^n$ в точку решітки $x' \in R^n$ і передає по каналу AWGN. Отриманий сигнал

$$y = rx' + z$$

де z – вектор гауссового шуму з дисперсією на вимір σ_z^2, та r – коефіцієнт загасання каналу.

СИСТЕМАТИЧНЕ ФОРМУВАННЯ ВОРОНОЇ

Для використання в якості коду каналу, що наближається до ємності для каналу AWGN, код решітки потребує двох елементів: решітки кодування з високим коефіцієнтом посилення кодування та методу формування, який задовольняє обмеження потужності з високим посиленням формування. Отже, у розд. III-A ми пропонуємо двоетапну конструкцію коду решітки, названу систематичним формуванням Вороного, для каналів AWGN, що призводить до хорошого кодування та покращення формування. Першим кроком є однозначне відображення інформаційних цілих чисел у розбиті цілі числа Вороного, які є точками всередині області Вороного формуючої решітки. Потім ми використовуємо підхід систематичного ґраткового кодування [15], щоб закодувати ці дизерингові цілі числа Вороного з використанням високовимірної решітки, так що кодові слова зберігають виграш формування від першого кроку та виграш кодування від високовимірної решітки. Потім ми обговорюємо альтернативний метод, для цілих чисел Вороного, використовуючи нерівномірні концепції сигналізації [18]. Далі ми обговорюємо двоетапну операцію декодування для систематичного формування Вороного. Нарешті, ми чисельно оцінюємо переваги у формуванні та кодуванні запропоновані схеми.

Систематичне формування Вороного: кодування

Спочатку ми опишемо цілі числа Вороного з дизерингом, метод для кодування цілих чисел у розбитих цілих числах, які знаходяться всередині фундаментальної області Вороного формуючої решітки. Це відображення є біективним. Ключова ідея тут полягає в тому, щоб сформувати відносно невеликі блоки цілих інформаційних чисел за допомогою низьковимірної решітки, а потім укласти їх у стопку, щоб утворити вектор високої розмірності, який потім кодується у високовимірну решітку. Спочатку ми опишемо властивості решіток кодування та формування, після чого наведемо кроки запропонованого відображення. Потім ми кодуємо ці з'єднані точки за допомогою високовимірної решітки.

Ми показуємо, що отримана конструкція коду наближається до того ж виграшу у формуванні та кодуванні, що й решітки формування та кодування.

Решітка кодування: Нехай $\Lambda_{c,n}$ — n-вимірна решітка кодування, визначена нижньою трикутною матрицею перевірки парності $H \in R^{nxn}$.

Нехай h_{ij} позначає (i, j)-ий елемент H. Нехай $\bar{H}$ — діагональна матриця nxn з i-им діагональним елементом, рівним h_{ii}, тобто $\bar{H}$ =diag(H). Розділимо діагональні елементи H на n/m груп, де m << n. Тоді будемо вважати, що для r-ї групи діагональні елементи рівні, тобто $h_{ii}=h_r$ для m(r-1)+1<=i<=rm.

Формування решітки: нехай $\Lambda_{s,m}$ — низька m-вимірна решітка, нехай $\theta \in R^{mxm}$ позначають її генераторну матрицю, нехай $V_{s,m}$ — її фундаментальна область Вороного, а P_m — її основний паралелепіпед. Генераторна матриця Θ повинна задовольняти кільком наступним властивостям. По-перше, Θ є нижньотрикутним. По-друге, діагональні записи Θ, масштабовані будь-яким діагональним елементом H, повинні бути цілими числами, тобто.

$$h_{jj}\theta_{ii} \in \mathbb{Z}, \quad \forall\ j = \{1, \dots n\}.$$

Нарешті, для кожного стовпця j

$$\theta_{ij}/\theta_{jj} \in \mathbb{Z}, \quad \forall\ i.$$

Добре відомі решітки, такі як D_m, E_8 і BW_{16}, масштабовані за $h_{jj}^1 M$, задовольняють цим умовам для $M \in Z$. Ці решітки мають хороші коефіцієнти формування, а також мають низькоскладні алгоритми декодування [19], що робить їх ідеальними для практичної реалізації.

2.1. Характеристика навчального курсу «Теорія інформації та кодування» (ТІК) в ДНУЗТ.

Предмет курсу «Теорія інформації та кодування» - передача інформації у повідомленнях. Але таке перенесення можна розглядати з різних сторін: наприклад, з погляду змісту інформації чи її призначення. У результаті ТІК такий розгляд проводиться з погляду ефективності передачі.

Таким чином, предметом дисципліни ТІК є методи забезпечення ефективності передачі повідомлень у технічному середовищі. При цьому ТІК розглядає передачу повідомлень із найзагальніших позицій (зокрема, у просторі та часі).

Невизначеність (ентропія) використовується для оцінки кількості інформації у ТІК. Наприклад, при передачі тексту на кожному кроці існує певна невизначеність щодо появи наступного символу. З появою того чи іншого знака така невизначеність усувається шляхом перетворення її на релевантну інформацію. І навпаки, коли знак передається по каналу, де може бути спотворений, канал додає невизначеності й те водночас віднімає частину переданої інформації.

Математичний апарат оцінки кількості інформації використовує теорію ймовірностей. Якщо конкретна подія має кілька варіантів, то її інформативність оцінюється через розподіл відповідних ймовірностей: чим більше варіантів і тим ближчий розподіл рівня події.

Такий підхід дає загальну основу перегляду основних завдань ТІК. Особливо:

- для компактності повідомлень необхідно виключити їхню крихкість, тобто передає тільки те, що є невизначеність (оскільки результат є прогнозом, він не несе інформації);

- точність буде забезпечена за рахунок протидії спотворенню каналу (а саме «шуму», який збільшує невизначеність і водночас зменшує обсяг інформації, що отримується). Зокрема, для цього потрібна передача додаткової «контрольної»

інформації;

- швидкість може бути збільшена, зокрема шляхом передачі символів з великою кількістю значень (наприклад, десятки цифр замість двійкових). Але ця передача стає вразливішою для «шуму». Математичний апарат дозволяє знайти оптимальні компроміси.

Таким чином, за рахунок використання математичного апарату для оцінки невизначеності ТІК має загальну основу для вирішення задач ефективності передачі. Далі ми докладно розглянемо відповідні математичні моделі та методи.

При передачі сигналів зазвичай вирішуються дві основні задачі:

- по-перше, необхідно використовувати форму сигналу, яка найкраще відповідає властивостям та обмеженням конкретного каналу (зокрема, типу фізичного середовища, обмеженням швидкості та характеристикам «шуму», а також необхідності одночасної передачі повідомлень з інших джерел);

- по-друге, необхідно знайти відповідний компроміс між двома принципово суперечать один одному показниками передачі - швидкістю передачі та її стійкістю до перешкод (визначає вірність). Справа в тому, що, збільшуючи швидкість, ми неминуче збільшуємо втрати енергії сигналу і, отже, його загасання. Відповідно, збільшується вплив шуму та збільшується ймовірність помилок розпізнавання.

Як і області кодування повідомлень, під час передачі сигналів виділяються окремі рівні, у яких вирішуються дуже специфічні завдання:

- у дискретній формі повідомлень (включаючи передачу двійкового коду) найбільше підходять імпульсні сигнали. Це саме те, що вони представляють на виході передавачів, включаючи мережеві карти або контролери зберігання даних. А імпульсні сигнали передаються в кабельному середовищі - по струмопровідних та оптичних лініях;

- для передачі в ефір (бездротова передача) потрібно використовувати радіохвилі як носій. При цьому застосування інформаційних імпульсних

сигналів на носії (модуляція) створює додаткові можливості як для узгодження з каналом (наприклад, виділення смуги частот, як у радіо), так і для регулювання співвідношення швидкості та точності (наприклад, при відносно низькому рівні шуму використовуйте більше інформативні сигнали з декількома фазами радіохвиль);

- поділ сигналів у середовищі відповідає зовнішньому ланцюзі передачі. У цьому випадку він може виконуватись як безпосередньо для імпульсних сигналів, так і для модульованих радіохвиль.

В ході ТІС ми познайомимося з теоретичними моделями (включаючи оцінку взаємодії сигналу з фізичним середовищем каналу, а також оцінку перешкоди сигналу та ефектів модуляції) і сучасними практичними рішеннями, які в даний час використовуються в комп'ютерних мережах і комунікаціях.

Мета та ідея ефективного кодування

Метою ефективного кодування є представлення масивів повідомлень з розміром алфавіту M компактними текстами, записаними кодовими словами, складеними із символів алфавіту меншої потужності:

$m < M$.

Оскільки порівнювати ефективність різних кодів з допомогою довжини закодованих послідовностей важко через відсутність представницьких вибірок масивів даних чи еталонного масиву (тексту), то оцінки ефективності використовується нормована величина, інваріантна до розміру тексту.

Таким чином, показником якості ефективного коду є середня довжина (точніше математичне очікування) кодового слова, що визначається як:

$$L = \sum_{j=1}^{M} P_j \cdot L_j,$$

де Pj – можливість отримання цього кодового слова;

Lj – довжина кодового слова і-го повідомлення;

M – кількість різних кодових слів (потужність алфавіту повідомлень).

Ідея «стиснення» тексту у тому, що найчастіші повідомлення кодуються короткими словами, а рідкісні – довгими. При цьому середня довжина кодового слова буде мінімальною. Зауважимо, що у природних мовами найчастіші слова – короткі, а рідкісні – довгі.

При стисненні інформації бажано обійтися без роздільників між кодовими словами, оскільки вони сильно подовжують закодовану послідовність. Для вирішення цього завдання кодові слова необхідно будувати так, щоб довші з них не починалися з коротких символів. Багато побудованих таким способом кодових слів, у яких всі приставки (префікси) різні, називається префіксним кодом. При цьому можлива однозначна дешифрація закодованого масиву, якщо відомий його початок, і в ньому немає спотворень окремих «розрядів» на заваді. Інакше помилкове читання «першого» чи «чергового» слова поширюється зазвичай кілька наступних слів. Таке помилкове читання, що неодноразово відновлюється, називається треком помилки.

Леон Георг Крафт (1949) встановив обмеження для існування безлічі слів розділеного префіксного коду потужністю M, записаних в алфавіті m, у вигляді:

$$\sum_{k=1}^{M} m^{-n_k} = \sum_{k=1}^{M} \frac{1}{m^{n_k}} \leq 1.$$

2.2. Підходи до удосконалення вивчення решіткового (trellis-) кодування в лабораторній роботі з курсу ТІК.

Відображення сигналів на переходи решітки

Унгербоек розробив евристичний набір правил присвоєння сигналів гілок переходів решітки для отримання ефективності кодування, який дозволяє зробити адекватний вибір станів решітки. Правила побудови решітки та розбиття множини сигналу (для модуляції 8-PSK) можна коротко викласти так.

1. Якщо за один інтервал модуляції кодується k біт, решітки повинні дозволяти для кожного стану 2k можливі переходи в наступний стан.

2. Між парою станів може існувати більше переходу.

3. Всі сигнали повинні з'являтися з рівною частотою і мати високу регулярність і симетрію.

4. Переходи з однаковим вихідним станом присвоюються сигналам або з підмножини B0 або B1- їх змішання неприпустимо.

5. Переходи з однаковим кінцевим станом присвоюються сигналам або з підмножини B0 або B1 - їх змішання неприпустимо.

6. Паралельні переходи присвоюються сигналам або з підмножини C0 або C1 або C2, або C3 - їх змішання неприпустимо.

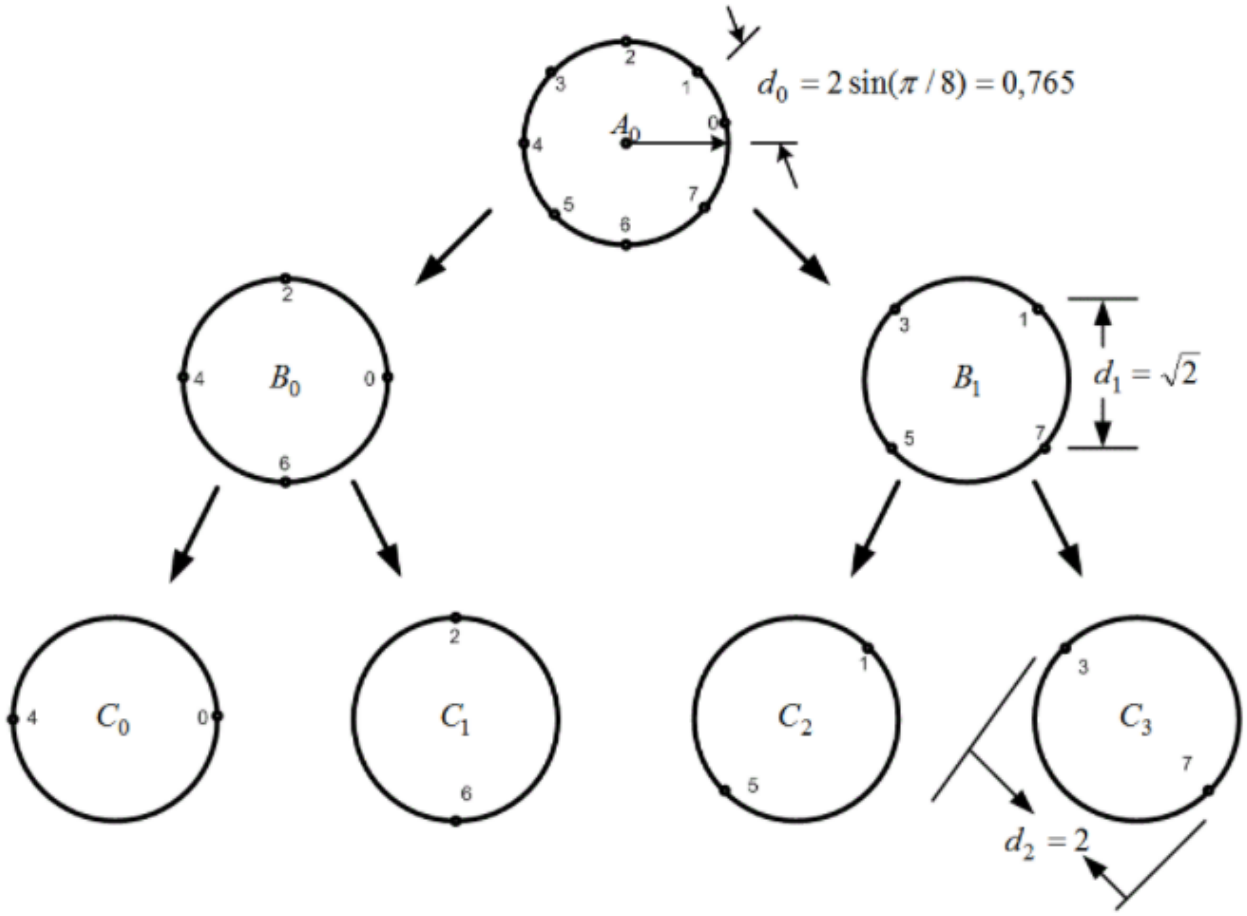

Рис. 2.1 - Розбиття Унгербоека набору сигналів 8-PSK

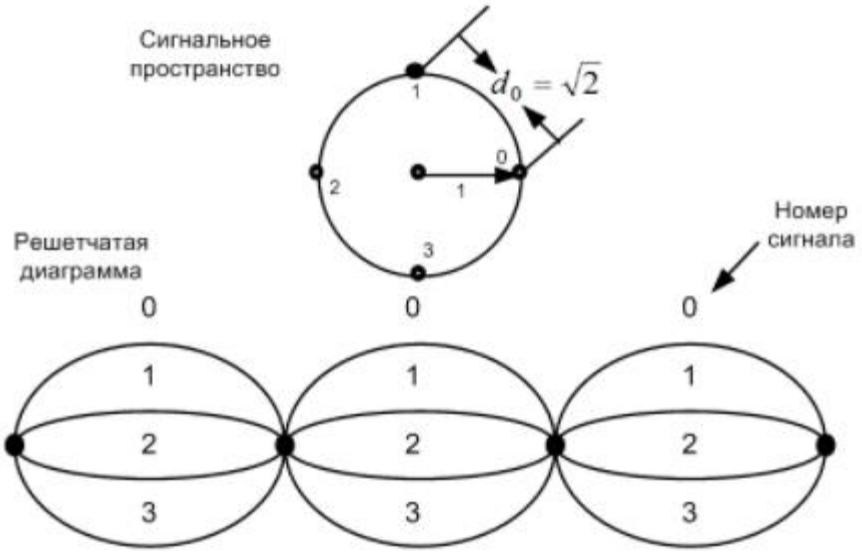

Рис. 2.2 - Некодована безліч сигналів 4-PSK та його решітчаста діаграма з одним станом.

Правила гарантують, що код, побудований таким чином, матиме регулярну

структуру та просвіт, який завжди перевищує мінімальну відстань між точками сигналу вихідної некодованої модуляції. На рис. 2.3 показано можливе відображення коду сигналу з використанням решітки з чотирма станами з паралельними шляхами. Присвоєння сигналу коду проводиться у вигляді вивчення розбитого простору сигналів (рис. 2.1), решітчастої діаграми, показаної на рис. 2.3, та правил, перерахованих вище.

На переходах ґрат написані номери сигналів, присвоєних цим переходам згідно з правилами розбиття. Зазначимо, що для модуляції 8-PSK присвоєння сигналу здійснювалося згідно з правилом 1: є k+1=3 кодових біта, отже k=2 інформаційних біта, а на вході та виході кожного стану є 22 = 4 переходи. Присвоєння сигналів здійснювалося згідно з правилом 6, оскільки кожній парі паралельних переходів було присвоєно сигнал одного з наборів C0, C1, C2 або C3. Крім того, присвоєння узгоджується з правилами 4 і 5, оскільки чотирьом гілкам, що виходять у стан (або залишає стан), були присвоєні сигнали набору B0або B1. На рис. 2.3 стани ґрат розрізняються згідно з типами сигналів, які можуть з'явитися на переходах, що залишають цей стан. Таким чином, стану можна позначити за допомогою підмножини сигналу як стан C0C1 або C2C3 або (інший можливий спосіб позначення за допомогою номерів сигналу) як стан 0426, 1537 і т.д.

На рис. 2.3 показані обидві системи позначень. З цього присвоєння модулюючих сигналів переходам у ґратах згідно з правилами розбиття випливає специфікація решітчастого кодера. Зазначимо, що остаточне присвоєння бітів коду сигналу (відображення кодового слова в перехід) можна виконати довільно. Хоча може здатися дещо дивним, що тепер можна безкарно привласнювати біти переходам у ґратах та сигналах, варто нагадати, що схеми кодера ще не існує. Отже, ще немає бітів і переходи у ґратах можуть мати лише той сенс, який для них виберемо ми. Які ж наслідки такого довільного присвоєння? Вибір різних відображень кодових слів у переходах позначиться на структурі кодера. Отже, якщо пощастить, буде реалізована схема кодера, вихідні біти якого відповідатимуть способу, яким здійснювалося їх привласнення переходів між

станами. В іншому випадку таке конструктивне рішення реалізувати буде складно. При деякому виборі способу присвоєння кодових слів конструкція кодера буде простіше, тоді як інший вибір може зумовити громіздкість його конструкції.

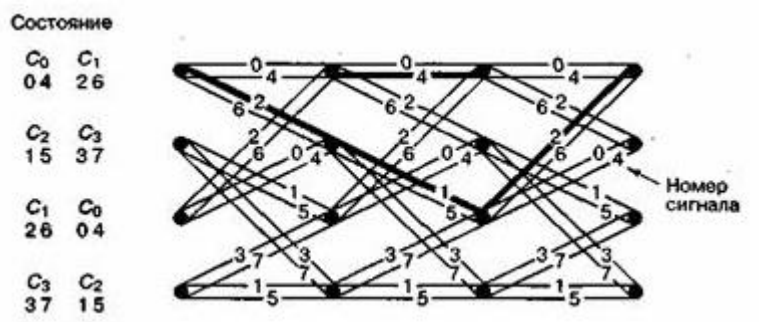

Рис. 2.3 - решітка з чотирма станами з паралельними шляхами.

Ґрати, аналогічні показаній на рис. 2.3 незабаром буде досліджено в контексті виявлення та декодування, щоб перевірити, чи забезпечується ефективність кодування при обліку в процесі кодування правил Унгербоека.

2.3. Аналіз існуючих аналогів лабораторних стендів для вивчення trellis-кодування.

Метод «вікна, що ковзає»

Найпоширеніший метод - це коли словник є «вікно», що переміщає текст повідомлення (рис. 2.4). З участю:

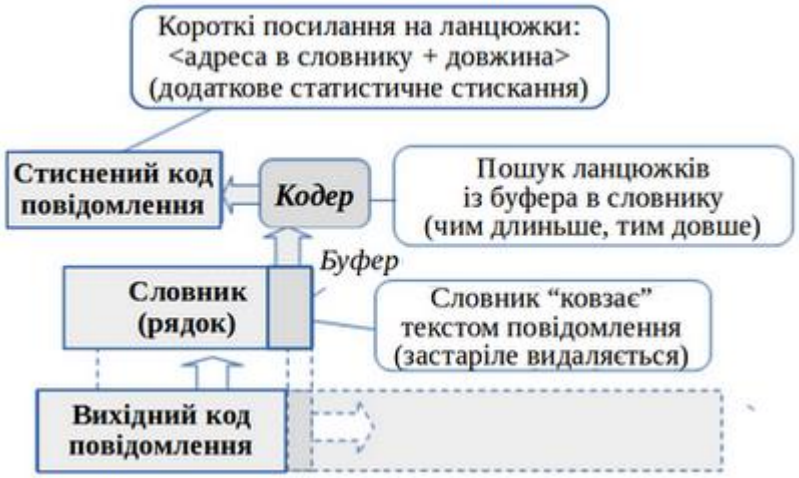

Рисунок 2.4 Схема стиснення зі «ковзаючим словником»

- Словник заповнюється при обробці повідомлення. Це схоже на переміщення рамки, яка переміщується від початку тексту до його кінця і всередину якої потрапляє кілька тисяч знаків (об'єм кількох сторінок книги). Пошук ланцюжків, що повторюються, ведеться тільки всередині кадру (Словник), а знаки, що виходять за його ліву межу, «забуваються». На початку кадру (біля правої межі) є вузький "слот" (так званий "буфер") для символів, коди яких підлягають стиску (зазвичай їхня кількість невелика - наприклад, від 16 до 64).);

- Кодувальник вибирає рядок символів із буфера та намагається знайти його у словнику. У разі успіху вміст буфера можна замінити коротким посиланням розділ словника. В іншому випадку кодувальник поступово вкорочує потрібний ланцюжок і повторює процедуру. Якщо у Словнику немає навіть коротких ланцюжків, кодувальник передає цю частину повідомлення окремими символами. У будь-якому випадку всі оброблені символи додаються до словника, при цьому символи «застарілі» звідти видаляються. У міру збільшення

обсягу словника ймовірність знаходження ланцюжків, що збігаються, збільшується, але час пошуку швидко збільшується;

- стислий код повідомлення міститься у файл архіву. Він включає посилання посилання, знайдені у словнику. Таке посилання формату [d, l] включає номер першого символу d (відстань) та довжину рядка l (довжину). Словник також включає стандартні коди символів для випадків, коли ланцюжки не виявляються у словнику. Щоб розрізняти ці два типи кодів, їм передує двійковий префікс (0 для коду знака та 1 для коду ланцюжка).

Розрахунок оновлення SISO з мінімальною решіткою

Будь-яке відображення без пам'яті, включаючи, наприклад, усі блочні кодери, може бути зручно представлено на належно визначеній решітковій діаграмі, точно як згортковий кодер. Основна відмінність по відношенню до решітчастої діаграми згорткового кодера полягає в тому, що в цьому випадку решітчаста діаграма змінюється в часі, тобто секції решітки змінюються в межах діаграми решітки. Більше того, будучи відображенням без пам'яті, набір початкових станів першої секції решітки і набір кінцевих станів останньої секції решітки мають потужність 1.

Діаграма решітки відповідає відображенню в тому сенсі, що всі дійсні конфігурації картографа відповідають шляху на решітковій діаграмі. Крім того, алгоритм SISO, для модуля SISO кодера решітки, може бути розширений для роботи також із змінюваними у часі діаграмами решітки, дозволяючи секціям решітки змінюватися в межах діаграми решітки. Складність алгоритму SISO пропорційна загальній кількості ребер у решітчастій діаграмі. Таким чином, складність буде мінімізована, якщо серед усіх можливих решітчастих уявлень розглянутого відображення знайти те, що пов'язане з мінімальною кількістю ребер (мінімальна решітка).

Ми розглядаємо як приклади два дуже важливі мінімальних решітки, які використовуються на практиці, тобто одне, пов'язане із загальним кодом повторення і те, яке пов'язане з його подвійним, кодом перевірки парності.

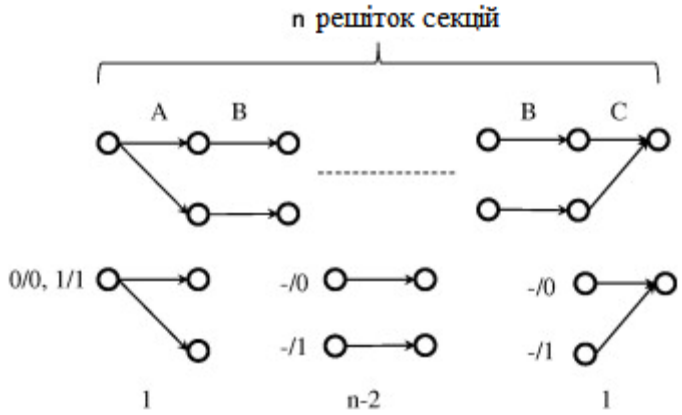

Рис. 2.5 - Мінімальна решітка діаграма, пов'язана із загальним (1,n) кодом повторення.

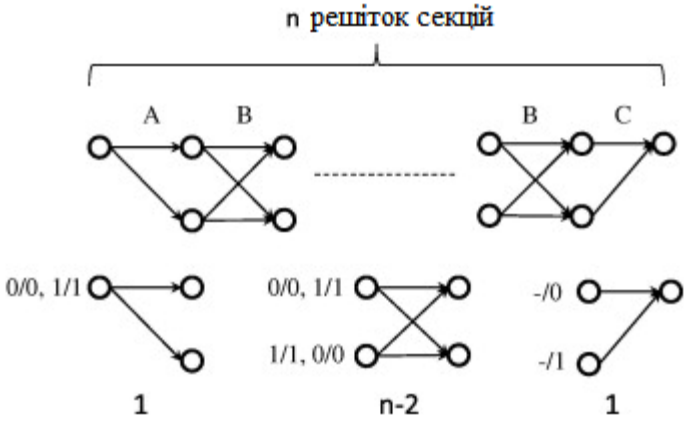

Рис. 2.6 - Мінімальна решітка діаграма, пов'язана із загальним (n-1,n) кодом перевірки парності.

Решеткова діаграма коду повторення складається з n секцій решітки трьох різних типів. Перша секція решітки (A) пов’язана з одним вхідним бітом і одним вихідним бітом, а наступні n-1 секції решітки (B) пов’язані з відсутністю вхідних бітів (1 ребро на стан) і одним вихідним бітом. Остаточна секція решітки (C) має єдиний кінцевий стан.

Загальна кількість шляхів дорівнює 2, що дорівнює кількості кодових слів. Складність, нормована на кількість розділів, становить:

$$C = \frac{\text{кількість ребер}}{\text{кількість секцій решіток}} = 2$$

і не залежить від розміру коду n.

Решеткова діаграма коду перевірки парності також складається з n секцій решітки трьох різних типів. Перша секція решітки, як і раніше, пов’язана з одним вхідним і одним вихідним бітом, а наступні n-1 секції решітки тепер мають

чотири ребра і пов'язані з одним вхідним і одним вихідним бітом. Остаточна секція решітки має єдиний кінцевий стан і не має вхідного біта. У цьому випадку загальна кількість шляхів решітки 2^{n-1} дорівнює кількості кодових слів. Тоді складність, нормована на кількість вхідних бітів, буде:

$$C \infty \frac{4n}{n} = 4$$

Також в цьому випадку він не залежить від розміру коду і набагато менший, ніж складність обчислення грубої сили, що пропорційно кількості кодових слів.

Фільтрація каналів і еквалайзери

На рис. 2.7 показана решітка, що характеризує прийнятий сигнал. Ця діаграма створена на основі рівняння з використанням чотирьох кроків.

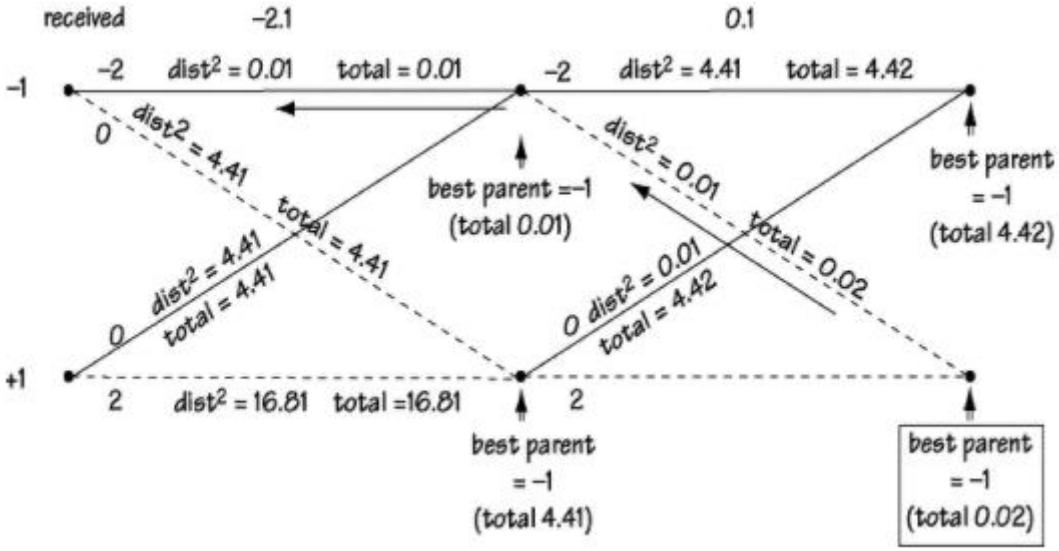

Рис. 2.7 - Решітка, що описує прийнятий сигнал + Алгоритм Вітербі, що використовується для отримання найкращого результату

На рис. 2.7 також показано застосування алгоритму Вітербі для визначення найкращого шляху через решітку. Для кожного вузла найкращий батьківський вузол визначається шляхом вибору батька з найменшою загальною відстанню. Коли ми дійдемо до останніх вузлів, ми шукаємо вузол з найменшою сумою і повертаємося назад через решітку, позначену стрілками. Ці стрілки говорять про те, що вихід шпалери має бути (-1 1) (на основі ліній розгалуження найкращого шляху).

Кодування та декодування каналів

Декодування каналу

Давайте тепер дослідимо, що робить декодер каналу. Спочатку визначимо завдання декодера каналу. На рис. 2.8 в канальний кодер надходить послідовність бітів m; ми розглянемо канальний кодер на рис. 2.8 з надходженням m = (0 0 0). Це відображається на виході, який ми назвемо u; у прикладі, який ми розглядаємо, u = (00 00 00). Ці шість бітів відправляються по каналу модулятором і повертаються до шести біт демодулятором. Біти, які виходять з демодулятора, які живлять декодер каналу, будуть називатися v. Ці біти v можуть дорівнювати або не дорівнювати u. Можливо, під час передачі сталася помилка, в результаті чого v = u + e, де e представляє бітову помилку. Для прикладу, який ми розглядаємо, ми скажемо, що отримуємо v = u + e = (00 00 00) + (00 00 01) = (00 00 01). У цьому випадку при передачі в позиції 6 сталася помилка одного біта.

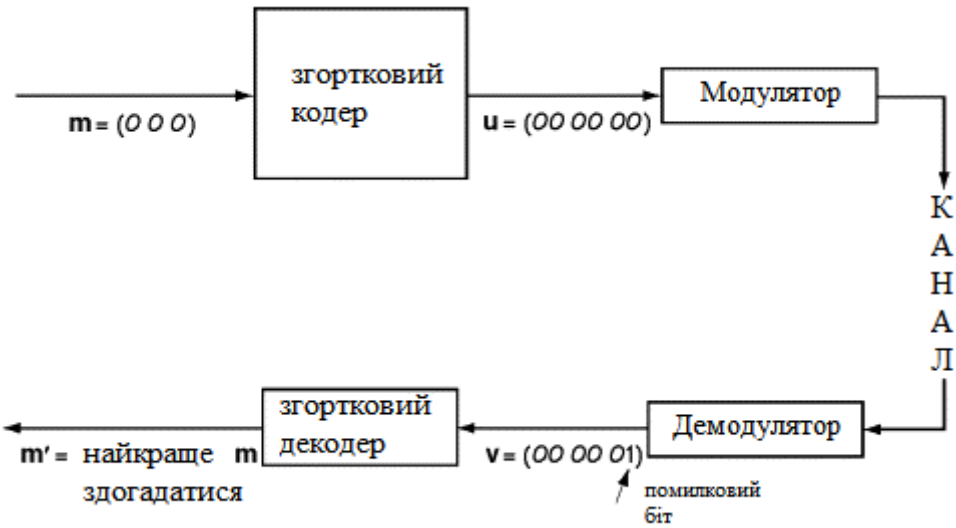

Рис. 2.8 - Роль канального (згорткового) декодера

Таким чином, мета канального декодера полягає в тому, щоб взяти біти v, які він отримав, і придумати найкраще припущення m, вихідних інформаційних бітів. Ми будемо називати припущення декодера каналу на m вектором m′. Ми хочемо знайти спосіб зробити m' якомога ближче до m.

Використання решітки

На решітковій діаграмі на рис. 2.8 показані можливі вихідні біти канального кодера. Якщо ви подивитеся на верхню частину пунктирних і пунктирних ліній решітчастої діаграми, ви побачите ці можливі результати.

Повернемося до нашого прикладу на 2.6 і розглянемо переданий вихід u = (00 00 00). Ми можемо використовувати діаграму решітки, щоб переконатися, що це u є можливим виходом кодера каналу. Дивлячись на діаграму решітки, ми бачимо, що (00 00 00) є можливим результатом, якщо слідувати ланцюжку з 00 виходів, які розташовані над суцільними лініями у верхній частині діаграми решітки. Якщо ви подивитеся на діаграму решітки, ви можете знайти вихід 00 в момент 1, а потім вихід 00 в момент 2, але ви побачите, що за ним не може слідувати вихід 01 в момент 3. Отже, отриманий v = (00 00 01) не є можливим виведенням канального кодера.

Таким чином, декодер каналу може зробити одне, це подивитися на отримане v (наприклад, v = (00 00 01)) і запитати себе, чи відповідає він вихідному шляху через решітку. Якщо відповідь негативна, то має бути помилка передачі. Це простий спосіб виявити помилки.

Згортковий декодер також може виправляти помилки за допомогою решітки. Для цього він просто запитує себе наступне: Враховуючи v = (00 00 01), який шлях у решітці є найближчим до цього v (з точки зору найменшої кількості різних бітів)? Вичерпний пошук усіх шляхів у решітці, найближчий шлях до v = (00 00 01) – це u′ = (00 00 00), як показано на рис.3.6. Канальний декодер вирішує, що надіслані біти повинні бути u´ = (00 00 00). «Якщо це надіслані біти, то вхід, який їх створив, дорівнює m´ = (0 0 0)», — каже декодер каналу, «і тому ми визначилися з нашим виходом».

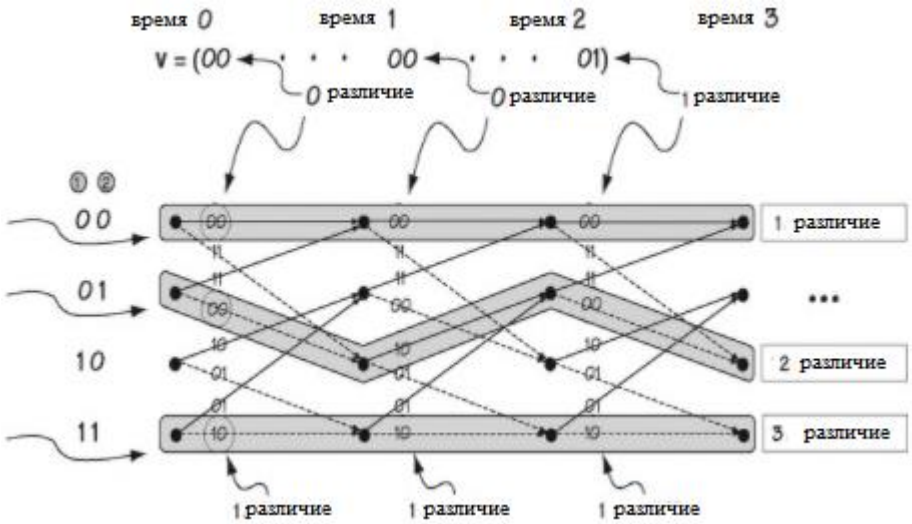

Рис. 2.9 - Вибір найближчого шляху в решітці (показано 3 шляхи)

Існує простий інструмент, який ми можемо використати, щоб знайти m′, коли ми визначили u′, найближчий шлях у решітці до отриманих бітів v. Все, що нам потрібно зробити, це подивитися на решітку – на ряд твердих і пунктирні лінії, що відповідає u´. Суцільна лінія говорить нам, що 0 — це біт у m′, а пунктирна лінія говорить нам, що 1 — це біт у m′. Тепер, якщо ви подивитеся на першу гілку шляху, ви побачите під 00 суцільну лінію, яка говорить вам, що перший вихідний біт в m´ є 0. Якщо ви подивитеся нижче другого 00, ви побачите другу суцільну лінію рядок – це означає, що другий вихідний біт дорівнює 0 і так далі.

Отже, функція згорткового декодера насправді досить проста. Він дивиться на v, потім дивиться на діаграму решітки й шукає її, поки не знайде вихідний шлях у решітки u´, найближчий до v. З цього u´ він приймає рішення та виводить m´.

3. РОЗРОБКА ПРОГРАМНОГО ЛАБОРАТОРНОГО СТЕНДА ДЛЯ ВИВЧЕННЯ

Для кодування повідомлення було реалізовано алгоритм згорткового кодування.

Опис згорткового алгоритму:

Для початку вибираємо параметри кодування. Вони вибираються один раз і будуть незмінними для всієї програми.

к - кількість біт у повідомленні, що передається на регістр;

n - кількість біт у закодованому повідомленні

K – довжина регістру;

$g : \{0,1\}^K \to \{0,1\}^n, g = g(k, n, K)$ - функція кодування (що залежить від попередніх параметрів);

$m \in \{0,1\}^N, m = m_1 m_2 ... m_N$ - кодове повідомлення (N може бути як завгодно великим).

Для зчитування даних було написана наступна частина коду

```
string s = msclr::interop::marshal_as<std::string>(textBox1->Text);
int k = atoi(msclr::interop::marshal_as<std::string>(textBox3->Text).c_str());
int n = atoi(msclr::interop::marshal_as<std::string>(textBox4->Text).c_str());
int K = atoi(msclr::interop::marshal_as<std::string>(textBox5->Text).c_str());
```

Для програмування алгоритму була написана наступна послідовність дій

```
string ans = "";
        for (int i = 0; i < K-1; ++i)
                s += '0';
        string str = "";
        for (int i = 0; i < K; ++i)
                str += "0";
```

```
for (int i = 0; i < s.size(); i += k)
{
    for (int j = 0; j < k; ++j)
        str = s[i + j] + str;
    string s2 = "";
    for (int j = 0; j < K; ++j)
        s2 += str[j];
    if (k == 1 && n == 2 && K == 3) {
        ans += to_string((int(s2[0]) - 48 + int(s2[1]) - 48 +
int(s2[2]) - 48) % 2);
        ans += to_string((int(s2[0]) - 48 + int(s2[2]) - 48) % 2);
    }
    if (k == 1 && n == 2 && K == 4) {
        ans += to_string((int(s2[0]) - 48 + int(s2[1]) - 48 +
int(s2[2]) - 48 + int(s2[3]) - 48) % 2);
        ans += to_string((int(s2[0]) - 48 + int(s2[2]) - 48 +
int(s2[2]) - 48) % 2);
    }
    if (k == 1 && n == 2 && K == 5) {
        ans += to_string((int(s2[0]) - 48 + int(s2[2]) - 48 +
int(s2[3]) - 48 + int(s2[4]) - 48) % 2);
        ans += to_string((int(s2[0]) - 48 + int(s2[1]) - 48 +
int(s2[4]) - 48) % 2);
    }
    if (k == 1 && n == 3 && K == 3) {
        ans += to_string((int(s2[0]) - 48 + int(s2[1]) - 48 +
int(s2[2]) - 48) % 2);
        ans += to_string((int(s2[0]) - 48 + int(s2[1]) - 48 +
int(s2[2]) - 48) % 2);
        ans += to_string((int(s2[0]) - 48 + int(s2[2]) - 48) % 2);
    }
```

```
                if (k == 1 && n == 3 && K == 4) {
                        ans += to_string((int(s2[0]) - 48 + int(s2[1]) - 48 +
int(s2[2]) - 48 + int(s2[3]) - 48) % 2);
                        ans += to_string((int(s2[0]) - 48 + int(s2[2]) - 48 +
int(s2[3]) - 48) % 2);
                        ans += to_string((int(s2[0]) - 48 + int(s2[1]) - 48 +
int(s2[2]) - 48) % 2);
                }
        }
```

Та для виведення результатів

```
textBox2->Text = gcnew System::String(ans.c_str());
```

Для створення trellis –модуляції було реалізовано наступну функцію

```
void trellis(unsigned char X, unsigned char Y,                  // Входы декодера
  unsigned char X0, unsigned char Y0,                   // Константы переходов
  unsigned char X1, unsigned char Y1,
  unsigned char M0_in, unsigned char M1_in,          // Входные метрики
  unsigned short int W0_in, unsigned short int W1_in,        // Входные пути

  unsigned char* out_bit0, unsigned char* out_bit1, // Выходные биты путей
  unsigned char* M0_out, unsigned char* M1_out,           // Выходные (новые)
метрики
  unsigned short int* W0_out, unsigned short int* W1_out  // Выходные пути
)

{

  unsigned char       delta_M0, delta_M1;       // Расстояние Хэмминга
  unsigned char       M_00, M_10, M_01, M_11; // Метрики переходов
```

```
// Вычисляем метрики для всех четырех переходов
delta_M0 = (X0 ^ X) + (Y0 ^ Y);
delta_M1 = (X1 ^ X) + (Y1 ^ Y);
M_00 = (M0_in + delta_M0);
M_10 = (M1_in + delta_M1);
M_01 = (M0_in + delta_M1);
M_11 = (M1_in + delta_M0);


// Для стануCond0_next выбираем меньшую метрику из двух путей, и запоминаем выживший путь
if (M_00 <= M_10)
{
        *M0_out = M_00;
        *W0_out = (W0_in >> 1) & 0x7FFF; // В регистр пути заносим ноль, соответствующий ребру решетки, идущему вверх
}
else
{
        *M0_out = M_10;
        *W0_out = (W1_in >> 1) & 0x7FFF;
}
*out_bit0 = (*W0_out & 0x0001);      // Запоминаем выходной бит регистра пути стануCond0_next


// Для стануCond1_next выбираем меньшую метрику из двух путей, и запоминаем выживший путь
```

```
    if (M_01 <= M_11)
    {
        *M1_out = M_01;
        *W1_out = (W0_in >> 1) | 0x8000; // В регистр пути заносим единицу, соответствующий ребру решетки, идущему вниз
    }
    else
    {
        *M1_out = M_11;
        *W1_out = (W1_in >> 1) | 0x8000;
    }
    *out_bit1 = (*W1_out & 0x0001); // Запоминаем выходной бит регистра пути стануCond1_next
}
```

Для encoder було розроблено наступну частину коду

```
void Encode(
    unsigned char in_bit,
    unsigned char* out_X, unsigned char* out_Y
)

{
    static unsigned char En1 = 0;
    static unsigned char En2 = 0;
    static unsigned char En3 = 0;
```

```
        *out_X = En1 ^ En3;
        *out_Y = En1 ^ En2 ^ En3;

        En3 = En2;
        En2 = En1;
        En1 = in_bit;
}
```

3.1.Аналіз та обґрунтування вимог до функціоналу та інтерфейсу користувача.

Коли програма завантажується вручну за допомогою перемикачів і запускається натисканням кнопки запуску. Однак наші комп'ютерні системи складні, і вони виросли за межами пропорції, створюючи великий семантичний розрив між додатками та обладнанням. Наприклад, остання операційна система (ОС) Microsoft XP має 40 мільйонів рядків коду. Настав час переглянути еволюцію обчислювальної техніки та шукати рішення, використовуючи нові парадигми. ОС і середовище швидко змінювалися з останніх 30 років, що робить речі застарілими, перш ніж вони зможуть стати продуктивними протягом свого життя. Наприклад, Microsoft випустила понад 20 основних випусків ОС за останні 25 років. Кожна мова та середовище випускають новий випуск кожні шість місяців. Коли ОС або її середовище змінюються, це має ефект пульсації, що призводить до застарілих програм. Як ми можемо зупинити таке поширення продуктів і технологій і зробити програми стабільними? Не забезпечуючи жодної перевірки, одним із можливих підходів є уникнення операційної системи та її середовища. У деяких областях обчислень це робиться повільно, непомітно.

Після завантаження ПК він перебуває в реальному режимі, де адресація обмежена 1 МБ і немає захисту вашого коду. Це просто в режимі дискової операційної системи (DOS). Щоб завантажувати та запускати більші програми та мати доступ до більшої пам'яті, вам потрібно навчитися переходити в захищений режим за допомогою інструкцій на мові асемблера. Однак усі виклики базової системи введення/виводу (BIOS) доступні лише в реальному режимі. Якщо ви хочете використовувати BIOS для вводу-виводу, вам потрібно мати механізми перемикання із захищеного режиму в реальний для виконання викликів BIOS. Таким чином, ваша програма на C++ або якась керуюча програма повинні виконувати це перемикання, прозоре для користувача. Якщо ви плануєте використовувати програмні переривання (їх 255) замість переривань BIOS, то вам потрібно написати власний код на зборці для вирішення всіх операцій введення-виводу. Ми використовували як BIOS, так і програмні переривання і

написали перемикач захищеного режиму в режим реального для роботи в обох режимах. Це нетривіальна річ, яку потрібно зробити під час складання, яка вимагає від вас ознайомлення з документом специфікації архітектури Intel, який доступний на їх веб-сайті. Цей документ є корисним ресурсом для розуміння та реалізації переривань, схем адресації, засобів завдань, пасток і винятків.

У звичайну програму C++ ми включаємо файли *.h, які є бібліотеками для ОС, які будуть включені до виконуваного файлу. Якщо ми використовуємо будь-які конструктори C++, вони автоматично включатимуть ці бібліотеки, які не мають сенсу в середовищі без ОС. Таким чином, середовище компіляції має використовувати пакетні файли для компіляції та зв'язування необхідних модулів. Як програміст прикладних програм на рівні мови C++, ми розробили файл інтерфейсу, який містить усі інтерфейси, необхідні для цих програм. Цей файл має API, включаючи: введення/виведення, завдання, пам'ять, таймер, спільну пам'ять, блокування тощо. Для створення виконуваних модулів ми використовуємо компілятор Visual Studio C++ (пакетний режим), ассемблер MASM 6.11 і компілятор Turbo. Ми написали всі пакетні файли для компіляції та компонування для завантаження, початкових програм і прикладних програм. Ці пакетні файли прості та легкі для розуміння та використання. API - це виклик C++, який, у свою чергу, викликає виклик C, а той, у свою чергу, викликає виклик збірки. Програміст C++ бачить лише API. Якщо інтерфейс можна створити на рівні C++, то код буде реалізовано на самому рівні C++. Ми уникаємо використання будь-яких прямих викликів збірки в програмі C++; однак такий виклик дозволений компілятором.

Архітектура

Архітектура програмного забезпечення дає пояснення того, як ваші системи поводяться на структурному рівні. Системи, які ви використовуєте, мають набір компонентів, розроблених для виконання певного завдання або набору завдань. Архітектура програмного забезпечення забезпечує фундамент, на якому все програмне забезпечення, яке є у компанії, можна змінити, створити або вилучити з експлуатації.

Архітектура програмного забезпечення впливає на якість, продуктивність, обслуговування та успіх системи на основі дизайну. Не розглядаючи архітектуру програмного забезпечення на регулярній основі, компанія відкриває себе для довгострокових наслідків, і проблеми, які можуть поставити їх системи під загрозу поломки, злому або низької продуктивності.

У сучасних системах існують загальні шаблони в архітектурі програмного забезпечення, які називаються архітектурними системами для програмного забезпечення. У більшості випадків для створення цілісної системи використовується кілька різних архітектурних систем, особливо для систем, які створювалися роками або працювали, або тих, які були побудовані різними розробниками.

У розроблюваній програмі необхідно, щоб була можливість задавати послідовність чисел параметрів кодування і мала можливість кодувати задану послідовність

3.2. Вибір програмних засобів для реалізації.

Середовище розробки - це набір процесів та інструментів, які використовуються для розробки вихідного програмного коду або програмного продукту. Це включає в себе все середовище, яке підтримує наскрізний процес, включаючи сервери розробки, підготовки та виробництва. Середовище розробки автоматизує або спрощує процедури, пов'язані зі створенням, тестуванням, налагодженням, встановленням виправлень, оновленням та супроводом програмного забезпечення, включаючи довгострокове обслуговування.

В рамках всеосяжного середовища розробки ви можете знайти інтегроване середовище розробки (IDE), яке є інструментом розробки, який розробники використовують для написання, складання, тестування та налагодження програмних програм. IDE надає зрозумілий та узгоджений інтерфейс, який підтримує процеси написання та тестування коду, а також його упаковки для випуску. Наявність єдиного інструменту для створення, зміни, компіляції, розгортання та налагодження програмного забезпечення скорочує час, необхідний для налаштування, коли команді доводиться збирати разом різноманітні утиліти розробки. Це також підвищує загальну продуктивність розробника та призводить до більш ефективного загального середовища розробки.

Написання безпечного коду, що забезпечує безпеку без уразливостей, є критичним викликом для кібербезпеки. Написання коду без уразливостей вже давно принаймні так само складно, як написання коду без помилок. Хоча в програмному забезпеченні є багато інших потенційних джерел ризику безпеки, розробка коду без відомих класів уразливостей завжди здавалася посильною метою. Він покладається на людей-розробників, які використовують інструменти, методи та процеси для створення програмного забезпечення, яке не має особливо відомих типів дефектів.

Один з найефективніших підходів — дослідження мов та інструментів

програмування — приніс технології, які, як показано, протистоять категоріям уразливостей, в основному, не допускаючи їх. Безпечні мови пам'яті, які керують виділенням та звільненням пам'яті, замість того, щоб вимагати від програміста це робити, унеможливлюють для розробників створення вразливостей переповнення буфера та деяких інших типів впливу через відсутні перевірки меж масиву, використання нульового покажчика та даних. витік через повторне використання пам'яті. Потокобезпечні мови можуть вирішувати випадки, коли умови гонки можуть бути використані, щоб підірвати перевірки, пов’язані з безпекою в програмі.

У межах спільноти розробників програмного забезпечення групи та організації, які мають на меті безпечну розробку програмного забезпечення, включили інструменти та методи у свій життєвий цикл розробки програмного забезпечення, щоб включити життєвий цикл безпечної розробки. Раннє програмне забезпечення з високою надійністю використовувало формальні методи для визначення властивостей безпеки системи, а також перегляд коду, щоб використовувати людину для пошуку таких недоліків на рівні кодування.2 Корпорація Майкрософт створила свій життєвий цикл розвитку безпеки, додавши аналіз першопричин, навчання безпеки, моделювання загроз, конкретні вимоги до безпечного кодування та тестування безпеки, яке включало тестування на проникнення та нечіткое тестування. Практика, як правило, впроваджується на основі потреб бізнесу, відчутного впливу на безпеку та відповідає усталеній або розвивається практиці розвитку.

Необхідні дослідження, які впливають на те, що працює, а що може працювати для безпечного розвитку. Здається, що нинішні дослідження відіграють, на жаль, обмежену роль у створенні, пропонуванні, оцінці та підтвердженні інструментів, методів і процесів, які використовуються на практиці для безпечного розвитку. Зокрема, дослідження рідко стосуються безпосередньо інструментів і методів, оскільки вони використовуються, у контексті, в якому вони використовуються. Нам потрібні додаткові дослідження ефективності та результатів безпечних інструментів, методів і процесів

розробки. Це дослідження можна судити про його вплив на те, як розробка програмного забезпечення працює на практиці. Властивості дослідження впливають на ймовірність такого впливу.

Суворість дослідницького наукового експериментування вимагає ряду вимог до процесу, включаючи формулювання гіпотези, що перевіряється, контроль змінних експерименту, щоб переконатися, що експеримент фактично перевіряє гіпотезу, і аналіз експериментальних даних і результатів для математичного підтвердження гіпотези (або спростувати нульову гіпотезу). Хоча ці процеси можуть стати основою важливих фундаментальних досліджень безпечної розробки, вони часто уникають безладних реалій, пов'язаних із впровадженням техніки на практиці, саме тому, що ці безладні реалії ускладнюють проектування експериментів.

Негативні результати дослідження, які не підтверджують, що безпечна техніка розробки підвищує безпеку, хоча й важливі для галузі досліджень, навряд чи вплинуть на безпечний розвиток на практиці. Перший урок розробника безпеки у великій технологічній компанії полягав у тому, що вказувати розробникам не робити чогось майже завжди було неефективним, якщо це не було поєднане з альтернативою, яку вони могли б використати для досягнення мети застарілої практики. Крім того, виявити, що інструмент або техніка експериментально неефективні для забезпечення безпеки, не доводить, що вони неефективні за межами контрольованого експерименту, у більш широкому, безладному та різноманітнішому контексті розробки програмного забезпечення.

Які з тих речей, які робляться в дослідженні, сподіваються на практичне перенесення в безпечний розвиток? Дві сучасні тенденції досліджень безпеки дають певну надію на безпечний розвиток. Одна з них полягає в тому, що безпечна розробка стала темою на конференціях з дослідження безпеки, охоплюючи такі теми, як оцінка інструментів, які допомагають розробникам уникати вразливостей, і вимірювання здатності розробників кодувати функціональні можливості, що стосуються безпеки. .

Інша обнадійлива тенденція — оцінка артефактів. Багато розробок програмного забезпечення базується на існуючому програмному забезпеченні, використовуючи фреймворки, бібліотеки та відкритий код. Пропонування артефакту, який використовується для встановлення та підтвердження ідеї дослідження, зменшує бар'єри для передачі цієї ідеї в розробку програмного забезпечення. Доступ до коду з відкритим кодом з умовами ліцензії, зручними для повторного використання, може збільшити його потенціал для використання. Деякі дослідницькі стимули змінюються, щоб заохочувати подання артефактів у рамках процесу подання та публікації наукової роботи на конференціях з безпеки, таких як USENIX Security і ACSAC.

Середовища розробки ПО (програмного забезпечення) є об'єднанням програмних засобів, які призначені для написання (створення) програмних продуктів. -Середовище розробки включає в свій зміст: компілятор, інтерпретатор, відладчик, засоби автоматизації збирання, а також редактор тексту.

Компілятор - це така програма, яка зчитує вихідні коди, написані програмістом і перетворює ці коди в програму.

Інтерпретатор - це програма яка зчитує команди, що знаходяться в вихідних кодах, відразу виконуючи їх.

Коли в середовищі розробки ПО присутні всі вищеназвані компоненти, тоді таке середовище називають інтегрованою. Такі середовища розробки збільшують темп, а також зручність розробки за рахунок: автоматизації, можливості виробляти весь цикл створення і розробки ПЗ.

Зазвичай середовище розробки ПО призначена для розробки тільки на одній мові програмування. А таке середовище розробки як інтегрована, надає право вибрати творцеві програми мову програмування для розробки, зручний розробнику (з мов підтримуваних цим середовищем). Прикладом цього є: Visual Studio, Komodo, Geany, Kylix, NetBeans, Eclipse.

MicrosoftVisualStudio - одна з інтегрованих середовищ розробки, розроблена на C ++ і C #, підтримується Windows OS. Дане середовище розробки переведена на десять мов (також і на російську мову). У Visual Studio творець може вести розробку веб-сайтів, веб-служб, писати консольні додатки, а також додатки з графічним інтерфейсом. Також VS підтримує різного роду доповнень. Найзнаменитіші доповнення - це Re Sharper (виконує пошук помилок в коді під час написання коду програми розробником, до компіляції); Visual Assist (на відміну від Re Sharper підтримує також і C ++); AnkhSVN (використовує в Visual Studio систему контролю версій, яка носить назву Subversion).

Переваги: Зрозумілий інтерфейс середовища розробки, зручність, автоматичне виявлення помилок в коді.

Недоліки: Складно для початківців програмістів.

Середовище особливо поширена в англомовних країнах, Росії, Китаї, Німеччині, Франції, Португалії, Італії, Японії, Іспанії та Кореї.

Geany також інтегроване середовище розробки програмного забезпечення. Підтримується на ОС Linux, а також на Mac Os і на Windows. Працює з тридцятьма двома мовами (також і з російською мовою). У складі Geany відсутня компілятор. Компілятор можна встановити як доповнення. Підтримує досить багато мов програмування, серед яких присутні класичний C. C ++ і C #.

Переваги: Простота і зручність, підсвічування вихідного коду, можливість підключати доповнення.

Недоліки: не включає до свого складу компілятор.

Серед розповсюджувалися в багатьох країнах (Більш ніж у тридцяти).

Komodo або ActiveStateKomodo - була написана на JavaScript, XUL, Python. Інтерфейс даного середовища тільки англійською мовою. Працює на тих операційних системах, як Geany: на Os Linux, Windows і Mac Os.

Підтримує десять мов програмування, серед яких присутні: PHP, Ruby, HTML5.

Переваги: Доповнення Code Explorer дозволяє переглядати об'єктне дерево скрипта або бібліотеки, середовище є кроссплатформовим, зручний відладчик з можливістю вилученого налагодження, можливість налаштувати інтерфейс середовища «під себе».

Недоліки: Висока вартість, підтримує мало мов програмування, сильно завантажує комп'ютер (а саме оперативну пам'ять), є складним для розуміння.

Поширена в основному в англомовних країнах.

Kylix - інтегроване середовище. Функціонує на OS Linux. Працює з C, C ++ і ObjectPascal.

В даному середовищі є можливість писати програми веб-служб.

Kylix випускався в трьох пакетах. Ці пакети: Enterprise Edition - включав в себе сто дев'яносто компонентів (був найбільшим і самим дорогим пакетом програми); Professional Edition (дешевший варіант, який включав в себе близько 165 компонентів); Open Edition - безкоштовний пакет програми, що містить в собі 75 компонентів, в ньому відсутній кошти для роботи з базами даних.

Оновлена версія Kylix 2, на відміну від Kylix працювала набагато швидше. Наприклад, Kylix 1 здійснював сортування бульбашкою масиву з 115 елементів півтори хвилини, Kylix 2 - одну секунду.

У 2002 році дане середовище розробки припинив підтримувати розробник.

Переваги: Зручний в перенесенні написаного з однієї операційної системи на іншу.

Недоліки: Дане середовище більше не підтримується розробником.

Поширена в основному в Європейських країнах і США, через те що

розробник (Borland) перестав підтримувати Kylix - стає все менш популярною і не затребуваною.

Netbeans - інтегроване середовище розробки програмного забезпечення. Була реалізована на програмному мовою Java. Це середовище розробки високої якості. Вміє працювати на декількох операційних системах, тобто є кроссплатформенной. Працює більш ніж з п'ятьма програмними мовами.

Переваги: Є безкоштовною, присутня система контролю версій, підсвічування синтаксису, можливо перейменовувати змінну / клас одним кліком, в тому випадку якщо вручну перейменовувати занадто довго (автоматизоване перейменування), є можливість форматування коду по CodeStyle, розробником середовище постійно вдосконалюється, поліпшується.

Недоліки: Часом в середовищі розробки виникають проблеми з кодуванням, довгий запуск програми.

Поширена в багатьох країнах, в силу того що є зручною і безкоштовною.

Eclipse - ще одна інтегрована середовище розробки ПО. Написана на мові Java в дві тисячі третьому році. Також є кроссплатформенной. За рахунок приєднуються до цього середовища доповнень - є можливість створювати програмні продукти більш ніж на п'яти мовах програмного коду.

Переваги: Постійне оновлення версій середовища розробки, підтримка багатьох мов (в тому числі і російського), є безкоштовною, підтримка багатьох мов програмування, середовище має промисловий рівень, є гнучкою - тобто легко налаштовується як під будь-яку платформу, так і під будь-якого користувача.

Недоліки: Сильно завантажує оперативну пам'ять комп'ютера, довго запускається, однак, якщо комп'ютер досить потужний - дана проблема легко вирішувана.

Поширена в багатьох країнах, користується популярністю.

1995 рік був особливо цікавим роком у світі комп'ютерного програмування. Саме в цьому році було випущено чотири нові мови програмування, які вплинули на світове співтовариство програмування таким чином, що не передбачалося на момент їх офіційного оголошення. До речі, це був також час, коли Інтернет тільки починав робити хвилі, а Інтернет був на межі вибуху в основну цифрову культуру.

Чотири мови, які дебютували в 1995 році, були Java, JavaScript, PHP і Ruby. Незважаючи на те, що їхні відповідні релізи не супроводжували особливої фанфари, ці мови з часом виростуть і стануть повсюдними інструментами програмування для більшості розробників програмного забезпечення. До цього часу домінуючими мовами були C і C++. Незважаючи на те, що ці двоє були дуже потужними, вони за своєю суттю не підходили для всесвітньої мережі. Крім того, для початківців програмістів (особливо C++) вони часто вважалися дещо складними та залякуючими.

Серед чотирьох, Java виявилася шаленим успіхом. З його часто цитованим гаслом «Напишіть один раз, запустіть будь-де» Java стала миттєвим хітом, оскільки її було набагато легше вивчити та опанувати (порівняно з C++). Java також представила нову ідею віртуальної машини (JVM), яка дозволила писати програми, які запускалися б на різних платформах без необхідності перекомпіляції.

В останні роки JavaScript витіснив Java як провідну мову програмування у світі. Постійне зростання JavaScript значною мірою пов'язано з впровадженням Node.js, технології, яка зробила можливим запуск JavaScript на стороні сервера.

PHP став домінуючою силою у сфері веб-програмування, особливо в поєднанні з іншими популярними технологіями з відкритим вихідним кодом, такими як Linux, Apache та MySQL. Разом вони утворили те, що зазвичай називають стеком LAMP.

Ruby здобув популярність серед веб-розробників після випуску шалено популярного веб-фреймворку Ruby on Rails у 2005 році датським програмістом Девідом Хайнемайєром Ханссоном (DHH).

Для великих (або командних) проектів в середу розробки повинні бути включені файловий менеджер, інтегроване середовище розробки програмного забезпечення, PlSql (використовується і для роботи з Системою Управління БД і як інструмент звітів), Cristal Reports (створення звітів), StarTeam (ведення журналу версій продукту, що розробляється).

Підводячи підсумки потрібно сказати про те, що інтегровані середовища розробки ПО дозволяють програмістам скоротити час на написання додатків, знизити затратність на написання (розробку), підвищити зручність розробки - що і є однією з основних цілей програмної інженерії.

Інтегровані середовища розробки зручні для командних проектів, остільки оскільки в таких середовищах можна виробляти весь цикл створення програмного забезпечення.

Інтегровані середовища зручні в написанні програм.

У роботі, нами була обрана середа розробки Visual Studio з використанням windows forms на мові програмування C++, так як на ній зручніше всього візуалізувати картину.

4. РОЗРОБКА МОДИФІКАЦІЇ МЕТОДИКИ ВИВЧЕННЯ TRELLIS-КОДУВАННЯ В КУРСІ ТІК

4.1.Рекомендації щодо змін методики вивчення trellis-кодування.

Як відомо, комп'ютер може обробляти та передавати інформацію у двійковому коді, тобто у вигляді послідовності логічних нулів та одиниць, званих бітами. Високий рівень напруги може бути узгоджений з логічним блоком, а низький рівень напруги може бути узгоджений з логічним нулем. При передачі інформації з телефонним проводам необхідно, щоб характеристики електричних сигналів, що передаються (потужність, спектральний склад і т. д.) відповідали вимогам приймального обладнання АТС. Одне з основних вимог - щоб спектр сигналу був у діапазоні від 300 до 3400 Гц, тобто. мав ширину трохи більше 3100 Гц. Щоб відповідати цьому та багатьом іншим вимогам, дані піддаються відповідному кодуванню, яким, власне, є модем. Існує кілька можливих методів кодування, при яких дані можуть передаватися по комутованих абонентських каналах.

Ці методи відрізняються один від одного як швидкістю передачі, так і стійкістю до перешкод. При цьому незалежно від методу кодування дані

передаються абонентськими каналами тільки в аналоговому вигляді. Це означає, що передачі інформації використовується синусоїдальний сигнал, який піддається аналогової модуляції. Використання аналогової модуляції призводить до значно меншого спектра з постійною швидкістю передачі. Аналогова модуляція - це метод фізичного кодування, при якому інформація кодується шляхом зміни амплітуди, частоти та фази сигналу синусоїдного несучої частоти.

Існує кілька основних способів аналогової модуляції: амплітудна, частотна та відносна фаза. Модеми використовують ці методи модуляції, але не окремо, а всі разом. Наприклад, амплітудна модуляція може використовуватися разом із фазовою модуляцією (амплітудно-фазова модуляція). Основна проблема, що виникає під час передачі інформації абонентськими каналами, - це збільшення швидкості. Швидкість обмежена смугою пропускання каналу зв'язку. Однак є спосіб значно збільшити швидкість передачі без збільшення ширини спектра сигналу. Основна ідея цього методу – використання багатопозиційного кодування. Послідовність біт даних розбивається на групи (символи), кожна з яких відповідає деякому дискретному стану сигналу.

Наприклад, використовуючи 16 різних станів сигналу (вони можуть відрізнятися один від одного як по амплітуді, так і фазі), ви можете кодувати всі можливі комбінації для послідовностей з 4 біт. Відповідно, 32 дискретних стану кодуватимуть групу з п'яти бітів в одному стані. На практиці збільшення швидкості передачі даних використовується в основному багатопозиційна амплітудно-фазова модуляція з декількома можливими значеннями рівнів амплітуди і фазового зсуву сигналу. Цей тип модуляції називається квадратурною амплітудною модуляцією (CAM). У випадку КАМ стан сигналу зручно відображати на площині сигнальної. Кожна точка площини сигналу має дві координати: амплітуду і фазу сигналу і є кодованою комбінацією бітів. Для підвищення завадостійкості квадратурної амплітудної модуляції може використовуватися так звана модуляція решітчастим кодом (PVL) або, альтернативно, решітчасте кодування.

При решітковій модуляції до кожної групи бітів, що передаються в одному дискретному стані сигналу, додається ще один надлишковий біт решітчастої діаграми. Якщо, наприклад, інформаційні біти розділені на групи по 4 біти (всього 16 різних комбінацій), то сигнальної площині міститься 16 сигнальних точок. Додавання п'ятого біта решітки призведе до 32 можливих комбінацій, тобто. кількість сигнальних точок подвоїться. Проте чи все бітові комбінації дозволені, т. е. значущі. Це ідея ґратчастого кодування. Значення доданого біта решітки визначається за спеціальним алгоритмом. Спеціальний кодувальник обчислює прикріплений біт ґрат.

На приймальному модемі для аналізу вхідних бітових послідовностей стоїть спеціальний декодер – так званий декодер Вітербо. Якщо прийняті послідовності дозволені, вважається, що передача безпомилкова, і біт ґрат просто видаляється. Якщо серед отриманих послідовностей є заборонені послідовності, то за допомогою спеціального алгоритму декодер Вітербо знаходить відповідну дозволену послідовність, виправляючи тим самим помилки передачі. Таким чином, сенс ґратчастого кодування - ціна відносно невеликої надмірності для підвищення стабільності передачі. Використання решітчастого кодування дозволяє в основному захистити від плутанини сусідні точки в сигнальному просторі, які найбільш схильні до можливості «змішування» під впливом перешкод.

4.2.Характеристика лабораторної роботи із вивчання кодування сигналів та її модуляції.

У цій роботі ми розглядаємо основні підходи передачі сигналів. Зокрема, розглянули можливість передачі інформаційних сигналів з використанням несучих сигналів (наприклад, радіохвиль). А також познайомилися з принципами розпізнавання сигналів на тлі шуму та оцінки їхньої перешкоди.

Основна мета лабораторної роботи:

-закріпити знання про інформаційну модель безперервного каналу з шумом та напрям узгодження параметрів сигналу та характеристик каналу;

-аналізувати підходи до узгодження спектра сигналу та смуги частот каналу для імпульсних та керованих сигналів;

-проаналізувати теоретичні підходи до оцінки завадостійкості основних видів маніпуляції сигналами та встановлення балансу швидкості передачі та перешкодозахищеності.

За рахунок модуляції можна вирішити низку завдань. Перший - це передача інформаційних сигналів у середовищі, не пристосованому при цьому (зокрема, з допомогою використання радіохвиль з'являється можливість передавати дискретні імпульсні сигнали великі відстані повітря). По-друге, модуляція дозволяє передавати спектр інформаційних сигналів у потрібній смузі частот (зокрема, так здійснюється радіопередача на хвилях різної довжини). По-третє, модуляція дозволяє підвищити стійкість інформаційних сигналів і захистити їх від несанкціонованого доступу (надалі ми познайомимося з конкретними рішеннями).

Існує два основні параметри класифікації типів модуляції радіосигналів.

- типи модуляції поділяються, зокрема, за параметром несучого

сигналу, який використовується для подачі інформаційного сигналу (для гармонійної несучої це може бути амплітуда, частота або фаза). Ще однією важливою ознакою класифікації є тип інформаційного сигналу – безперервний або дискретний (для безперервного інформаційного сигналу безпосередньо використовується термін «модуляція», а для дискретного – термін «маніпуляція»);

- Основні типи модуляції – стандартні скорочення, які часто використовуються у технічних текстах. Зокрема, безперервна модуляція (модуляція) амплітуди (амплітуда), частоти (частоти) та фази (фази) позначається як AM (AM), FM (FM) та FM (PM), а відповідні типи дискретних маніпуляцій (зсув -keying) - як Amn (ASK), CMN (FSK) та FMN (PSK). Оскільки безперервна зміна частоти та фази сигналу безпосередньо пов'язана, ці два типи модуляції часто поєднуються під загальною назвою «кутова» (FM-PM). Всі види модуляції використовуються відповідно до їх особливостей, які ми розглянемо нижче.

4.3. Обґрунтування програми, що була розроблена.

Основуючись на згортковому алгоритмі було розроблено наступну програму, вигляд якої зображено на рис. 3.1

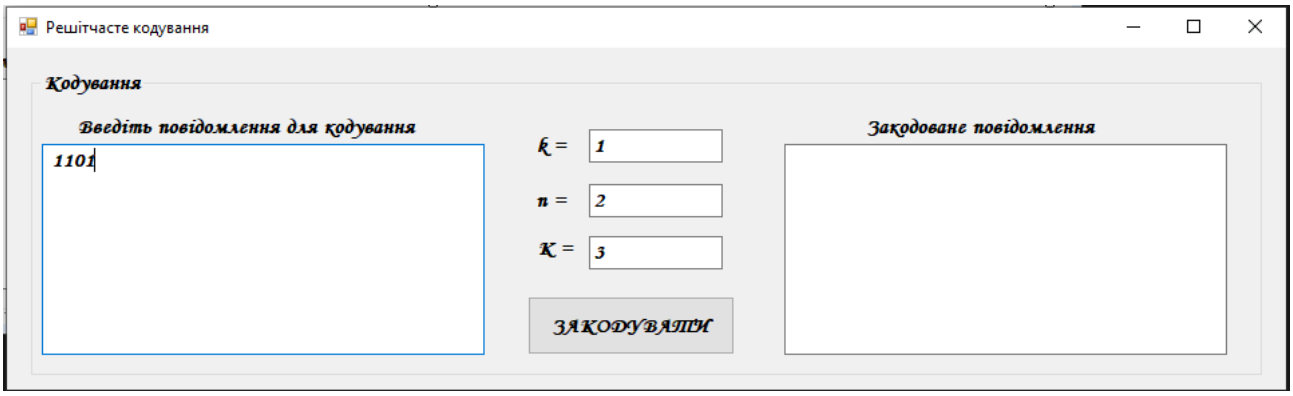

Рис. 3.1 – Програма кодування повідомлення

В ній є поле для введення повідомлення для кодування

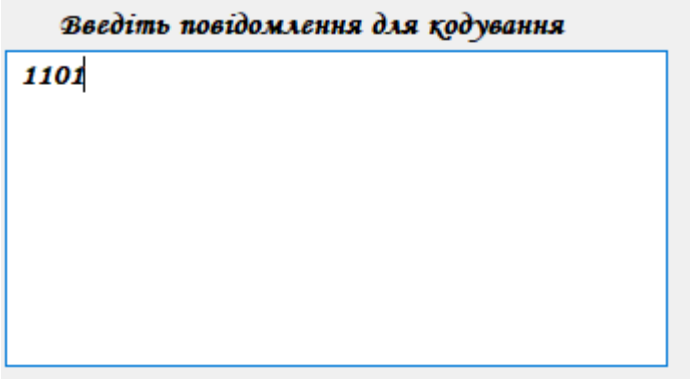

Рис. 3.2 – Поле для введення повідомлення

Поля для введення параметрів кодування

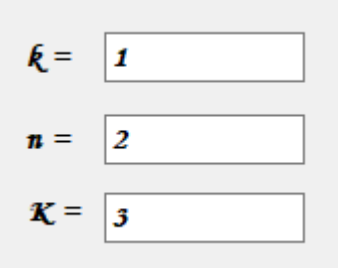

Рис. 3.3 – Параметри кодування

Поле у яке потрапить закодоване повідомлення

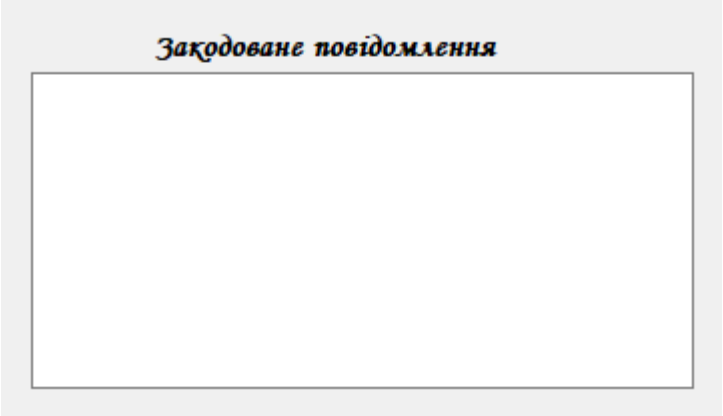

Рис. 3.4 – Поле для закодованого повідомлення

Та кнопка закодувати

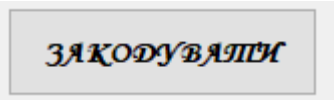

Рис. 3.5 – Кнопка «Закодувати»

Нижче наведено декілька результатів роботи програми

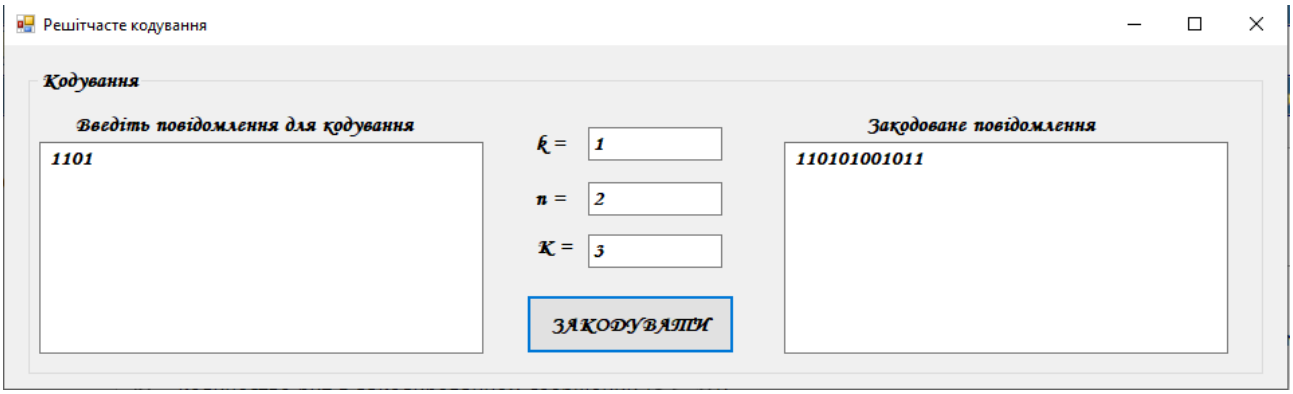

Рис. 3.6 – Результат роботи програми

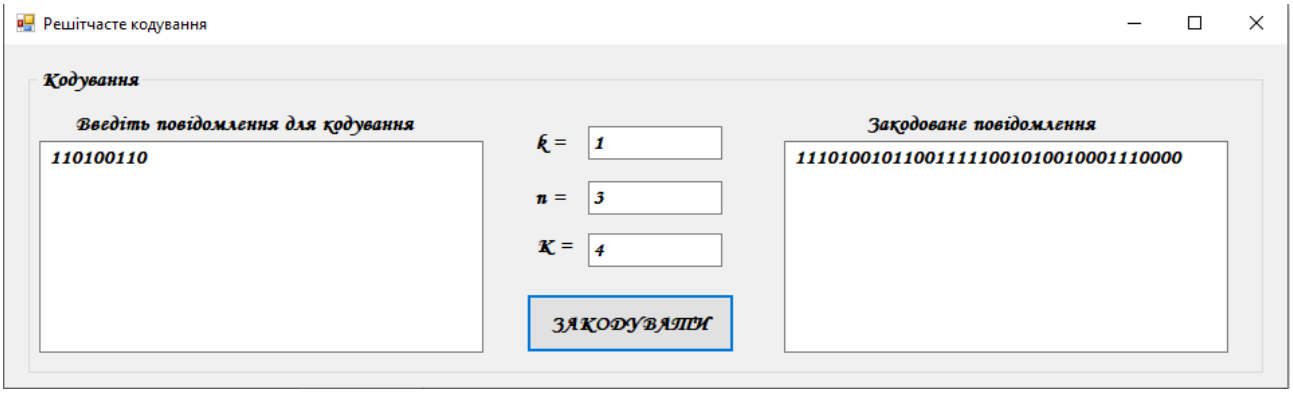

Рис. 3.7 – Результат роботи програми

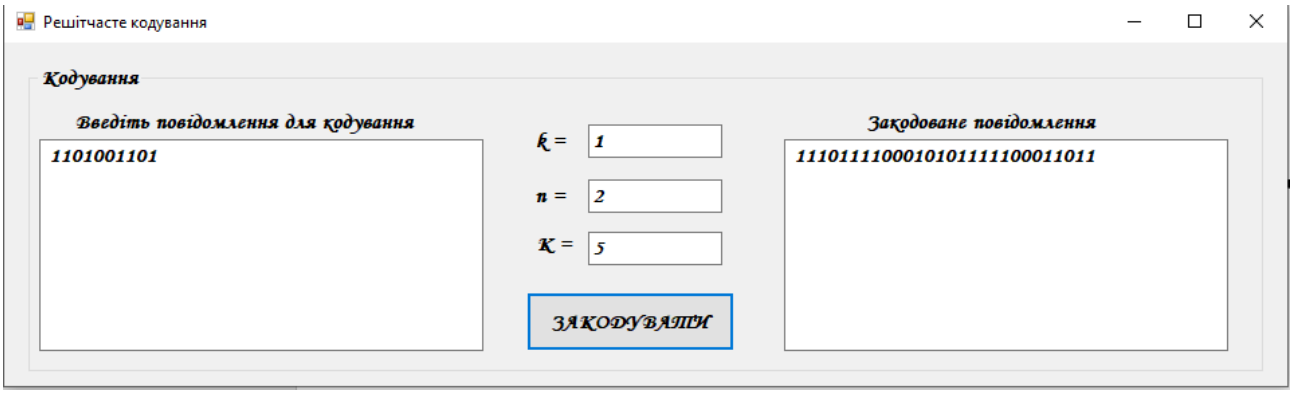

Рис. 3.8 – Результат роботи програми

Висновки

У роботі представлено нове рішення наукової проблеми, яке полягає у

забезпеченні достовірності інформації шляхом розробки методів, що ґрунтуються на адаптивному кодуванні, які разом утворюють нову інформаційну технологію для забезпечення достовірності інформації. Основні результати роботи:

1. Аналіз існуючих методів забезпечення достовірності інформації в умовах апріорної невизначеності, аналіз поточного стану, який показав, що перспективним напрямом вирішення проблеми забезпечення достовірності інформації в умовах апріорної невизначеності є адаптація коду.

2. Метод забезпечення достовірності інформації, заснований на адаптації різних структур коду, що дозволяє забезпечити задані показники надійності інформації та скоротити кількість основних операцій цифрових сигнальних процесорів під час цифрової обробки кодованих даних.

Відмінність розробленого методу від існуючих, що визначає його новизну, полягає у використанні різної структури шумостійких кодів, від більш простих до більш складних, залежно від відношення сигнал/шум у каналі, що призводить до цифрових сигнальних процесорів при цифровій обробці закодованих даних на 39-46. % у разі нестаціонарних навмисних перешкод за зміни коефіцієнта перешкодних сигналів від 6 до 1 дБ.

Використання методу з допомогою запровадження функцій власності дозволяє забезпечити задані показники достовірності інформації та підвищити енергоефективність на 0,2-0,8 дБ проти відомими методами.

5. ЗАБЕЗПЕЧЕННЯ ДОСТОВІРНОСТІ ІНФОРМАЦІЇ ТА ПЕРЕВІРКА НА ПЛАГІАТ

5.1. Етапи забезпечення достовірності інформації

Нині методи забезпечення достовірності інформації глибоко та широко досліджуються у наукових працях вітчизняних та зарубіжних авторів, серед яких найбільш відомими є такі вчені: А.Г. Зюко, Д.Д. Кловський, М.Л. Теплов, Л.М. Фінк, Л.Є. Варакін, В.Л. Банкет, В.В. Квашенников, В.І. А. Голдсміт, М. Валенті та інші.

Проте такі завдання вивчені недостатньо і потребують додаткового вивчення:

- створення нових та вдосконалення існуючих методів забезпечення достовірності інформації у БЗПД;
- розробка, ефективне використання обчислювальних методів для вирішення завдань надійності, проектування, виготовлення та експлуатації нового обладнання та нових технологій;
- модифікація та спеціалізація існуючих обчислювальних методів з метою підвищення їх ефективності, створення та дослідження нових обчислювальних методів та алгоритмів, що враховують особливості БЗПД;
- розробка ефективних методів адаптивного шумостійкого кодування для забезпечення заданих характеристик надійності інформації в каналах з

високим рівнем шуму та перешкод з урахуванням нечітких правил прийняття рішень;

- у системах з параметричною адаптацією у разі збільшення рівнів шуму та перешкод до певного рівня забезпечення заданих характеристик достовірності інформації стає неможливим, тому виникає питання про багаторівневу адаптацію не лише параметрів, а й структури коду.

Таким чином, дисертація вирішує актуальне наукове та прикладне завдання, що має важливу наукову, практичну та технічну спрямованість у створенні та забезпеченні достовірності інформації шляхом розробки методів, заснованих на адаптивному кодуванні.

Об'єктом дослідження у дисертації є процеси формування та обробки кодованих даних при бездротовій передачі даних, а предметом дослідження – методи забезпечення достовірності інформації при бездротовій передачі даних.

Таким чином, метою дисертації є підвищення ефективності роботи в умовах перешкод за рахунок забезпечення достовірності інформації за рахунок розробки методів, що базуються на адаптивному кодуванні, та їх використання.

Вирішення сформульованого науково-прикладного завдання має здійснюватися поетапно.

У першому етапі необхідно формалізувати процес адаптивного зміни структури коду.

З другого краю етапі необхідно розробити метод забезпечення достовірності інформації, заснований адаптації структур коду.

На третьому етапі необхідно розробити та реалізувати у програмній формі обчислювальний метод нечіткого декодування.

На четвертому етапі необхідно розробити методику підготовки первинної інформації адаптивних кодів.

На п'ятому етапі необхідно проаналізувати отримані результати порівняння

з відомими методами моделювання.

Список літератури

1. Ю. Бірк і Т. Кол, “Кодування з інформованим джерелом за запитом по каналах мовлення”,. Спільна конф. Комп'ютерні та комунікаційні товариства IEEE (INFOCOM), том. 3 березня 1998 р., с. 1257–1264.

2. З. Бар-Йоссеф, Ю. Бірк, Т. С. Джайрам і Т. Кол, “Індексне кодування з побічною інформацією”, Інф. Теорія, вип. 57, № 3, с. 1479–1494, березень 2017 р.

3. Н. Алон, Е. Любецький, У. Став, А. Вайнштейн, А. Хасідім, “Віщання з побічною інформацією”, в 49-й Основи комп’ютерних наук, жовтень 2008 р., с. 823–832.

4. С. Ель Руайхеб, А. Спринтсон та К. Джорджіадес, “Про проблему кодування індексу та її зв’язок з мережевим кодуванням і теорією

матроїдів", Інф. Теорія, вип. 56, № 7, с. 3187–3195, липень 2010 р.

5. А. Бласяк, Р. Д. Кляйнберг та Е. Любецький, «Індексне кодування за допомогою лінійного програмування», препринт, 2019. [Онлайн]. Доступно: http://arxiv.org/abs/1004.1379

6. С. Унал та А. Вагнер, "Загальне індексне кодування з побічною інформацією: регістр трьох декодерів", Симп. Теорія інформації, липень 2018 р., с. 1137–1141.

7. С. В. Тюлюпа, В. С. Наконечний та Л. О. Комарова, " Системи управління, навігації та зв'язку ", Симп. Побудова сигнально-кодових конструкцій для багатопроменевих каналів спеціального радіозв'язку, 2014 р., випуск 2(30), с. 158.

8. Г. Крамер та С. Шамай, «Пропускна здатність для класів мовних каналів із побічною інформацією приймача», у Семінар із теорії інформації, вересень 2007 р., с. 313–318.

9. Л.-Л. Хіе, «Мережне кодування та випадкове об'єднання для багатокористувацьких каналів», 10-й канадський семінар з теорії інформації, червень 2007 р., с. 85–88.

10. Кузьмин І. В. Основи теорії інформації і кодування / І. В. Кузьмин, В. А. Кедру. – К. : Вища школа, 2003. – 220 с.

11. Лукін А. Ведення в цифрову обработку сигналів (Математичні основи) / Лукін А. А. – М. : МГУ, Лабораторія комп'ютерної графіки та мультимедіа, 2002. – 616 с.

12. Дж. Сіма та В. Чен, «Спільна мережа та кодування Гельфанда-Пінкера для 3-приймальних гауссових каналів мовлення з побічною інформацією приймача», Теорія інформації, червень 2014 р., с. 81–85.

13. Б. Асаді, Л. Онг та С. Джонсон, «Пропускна здатність каналів широкомовної трансляції AWGN з трьома приймачами з побічною інформацією приймача», Теорія інформації , червень 2014 р., с. 2899–2903.

14. Є. Тунсель, "Кодування Слепяна-Вовка по каналах мовлення", IEEE Trans. Інф. Теорія, вип. 52, № 4, с. 1469–1482, квітень 2006 р.

15. Л. Сяо, Т. Фуджа, Дж. Клівер та Д. Костелло, «Вкладені коди з множинними інтерпретаціями», 40-річчя. конф. Інформаційні науки та системи (CISS), березень 2006 р., с. 851–856.

16. Y. Ю. Ма, З. Лінь, Х. Чен та Б. Вучетіч "Множина інтерпретацій для кодованих систем бездротової ретрансляційної мережі з багатьма джерелами," в Симп. Персональний внутрішній та мобільний радіозв'язок, вересень 2012 р., с. 2253–2258.

17. Ф. Барбоза та М. Коста, "Метод побудови дерева вкладених циклічних кодів", Семінар з теорії інформації, жовтень 2011 р., с. 302–305.

18. Y.-C. Хуанг, "Коди решітки з полів алгебраїчних чисел",. Теорія інформації, червень 2015 р., с. 2485–2489.

19. Л. Натараджан, Ю. Хонг та Е. Вітербо, "Індексні коди для гаусового каналу мовлення з використанням квадратурної амплітудної модуляці. 2020р.

20. Брайловский, И.В. Новый метод частичных обобщенных интервальных преобразований кодирования источников без памяти / И.В. Брайловский // Чебышевский сборник. – М. : Изд-во МГУ. 2003. Т. 4. Вып. 4. – С. 36–47.

21. Гуменюк, А.С. О длине двоичных слов для кодирования интервалов строя знаковой цепи // Информационные технологии и математическое моделирование (ИТММ-2005) : Матер. IV Всеросс. науч.-техн. конф. – Томск : Изд-во Том. унт-та, 2005. Ч. 1. – С. 44–46.

22. **Темников, Ф.Е. Теоретические основы информационной техники / Ф. Е. Темников, В.А. Афонин, В.И. Дмитриев. – 2-е изд., испр. и доп. – М. : Энергия, 1979. – 512 с.**

23. Дмитриев, В.И. Прикладная теория информации : учеб. для студентов / В.И. Дмитриев. – М. : Высш.шк., 1989. – 320 с.

Додаток В

ДОСЛІДЖЕННЯ ТА РОЗРОБКА ЗАСОБІВ ВИВЧЕННЯ РЕШІТКОВОГО КОДУВАННЯ

Устенко А.Б., Нікітін В.М. к.т.н., доцент, доцент кафедри ЕОМ
Український державний університет науки і технологій

«Решіткове кодування» або “решітково-кодована модуляція” (Trellis Coded Modulation — TCM) є важливим компонентом технологій передачі повідомлень каналами із шумом. Зокрема аналіз його сучасних застосувань свідчить, що найбільш поширеним тут є стандарт SHDSL, який орієнтується на передачу даних кабелями із вузькою частотною смугою. Рішення на базі SHDSL нині використовуються зокрема в промисловості. Вони мають певні переваги щодо захищеності (порівняно з передачею в ефірі) та економічності (порівняно із використанням оптичних кабелів). Таким чином, ця технологія, хоч і не належить до найбільш поширених, має певні сфери сучасного використання.

Нині студенти кафедри ЕОМ знайомляться із технологією TCM в межах навчального курсу “Теорія інформації та кодування”. Цінність її вивчення обумовлена синтетичним характером TCM: ця технологія поєднує елементи збиткового (зокрема “згорткового”) кодування і сучасних видів модуляції (зокрема PSK та QAM). Тож її вивчення дає розуміння, як саме можна конвертувати збитковість кодування в завадостійкість сигналів. До того ж TCM використовує алгоритм декодування Вітербі, який є найбільш поширеним в царині згорткових кодів, і його детальний розгляд на цьому прикладі є досить доречним.

Аналіз підходів до викладання засад TCM в сучасній версії курсу “Теорія інформації та кодування” свідчить, що воно є досить узагальненим і не завжди забезпечує повне розуміння принципів технології, зокрема складної взаємодії схеми кодера та модулятора. Це зокрема пов'язано із обмеженням часу на вивчення широкого класу кодів із захистом від помилок. Такий недолік можна компенсувати поглибленим вивченням в лабораторній роботі із використанням відповідної демо-програми.

Пошук в Інтернет засобів, що наочно ілюструють дію технології TCM, виявив значну кількість якісного графічного матеріала, а також демонстрації, які відображають певні етапи технології в динаміці. Однак завданням лабораторної роботи повинно бути самостійне дослідження, що виконується студентом згідно індивідуальному варіанту. Отже створення оригінальної демо-програми, яка б відповідала цим вимогам, є актуальним.

За результатами аналізу запропоновані вимоги до програми демонстрації засад TCM:

- відобразити динаміку роботи регістра-кодера, що перетворює вхідний потік інформаційних бітів в подвоєний потік “дібітів”, які надалі візьмуть участь в формуванні завадостійких сигналів;
- відобразити динаміку змін стану схеми в наочній формі решіткової діаграми. Показати на прикладах етапи кодування та декодування із виявленням помилок за алгоритмом Вітербі;
- відобразити конверсію одержаних кодових послідовностей в сигнали PSK/QAM, наочно показавши переваги способу щодо завадостійкості сигналів, в тому числі із відповдною оцінкою в децибелах.

В дипломній роботі передбачається також розробка тестових питань щодо вивчення технології TCM, а також розробка методики застосування програми і аналіз її практичного застосування студентами.

Додаток Г

Додаток Г – Довідка про порушення академічної доброчесності

Довідка

про відсутність плагіату у випускній кваліфікаційній роботі

Міністерство освіти і науки України

Український національний університет науки і технологій

Кафедра Електронних обчислювальних машин

ДОВІДКА

За результатами перевірки випускної кваліфікаційної роботи здобувача вищої освіти Нікітіна Вячеслава Миколайовича на тему: “Дослідження та розробка засобів вивчення решіткового кодування” в роботі не виявлено порушень академічної доброчесності.

Керівник ВКР доцент кафедри ЕОМ

_____ Устенко А.Б.

Додаток Д

Додаток Д – Акт перевірки на плагіат ВКР

Додаток Е

Додаток Е – Заява о перевірці на плагіат

ЗАЯВА

Я, Нікітін Вячеслав Миколайович_____

(ПІБ повністю)

Студент групи КС2021_____

(шифр групи)

Спеціальності 123 Комп'ютерні системи та мережі_____

(код та назва спеціальності)

освітньої програми Комп'ютерна інженерія _____

(назва освітньої програми)

освітнього ступеня підготовки магістр_____

(бакалавр, магістр)

Заявляю, що моя випускна кваліфікаційна робота на тему:

Дослідження та розробка засобів вивчення решіткового кодування_____

виконана самостійно і в ній не міститься елементів плагіату. Всі запозичення з друкованих та електронних джерел мають відповідні посилання. Прошу перевірити її на наявність академічного плагіату.

Я ознайомлений(а) з чинним «Порядком перевірки кваліфікаційних випускних робіт здобувачів вищої освіти на виявлення текстових та графічних запозичень засобами перевірки на плагіат», згідно з яким виявлення плагіату є підставою для відмови в допуску випускної кваліфікаційної роботи до захисту.

Дата Підпис_____

Керівник: Устенко А. Б. Підпис_____

(ПІБ)