ТЕМА 8. МЕТОДИ І ЗАСОБИ ОБРОБКИ ІНФОРМАЦІЇ

8.1. Економне кодування. Префіксний код та його дерево. Алгоритм Шеннона-Фано. Алгоритм Хаффмана. Основні методи економного кодування без втрат послідовної дискретної інформації

8.2. Принципи завадостійкого кодування. Основні характеристики завадостійких кодів. Класифікація завадостійких кодів. Математичний опис процесу кодування і декодування

8.3. Блочні лінійні коди. Коректуючи властивості блочних кодів. Коди з перевіркою на парність. Коди Хеммінга

8.4. Циклічні коди. Засоби опису циклічних кодів. Властивості циклічних кодів по виявленню помилок. Укорочені циклічні коди. Коди Боуза-Чоудхурі-Хоквінгема. Коди Ріда-Соломона. Код Файра

8.1. Економне кодування

Спосіб одержання компактного подання інформації зветься економне кодування.

Теорія економного кодування поєднує в собі кілька різних **напрямків**. У рамках даної теорії методи прийнято розділяти на методи **економного кодування інформації без втрат** (*lossless*) і методи **економного кодування інформації з втратами** (*lossy*). Як слідує з назв, обробка інформації методами першої групи не веде до інформаційних втрат, тоді як використання методів другої групи сполучено з такими втратами.

Методи останньої групи застосовуються до інформації, що містить окремі несуттєві складові (за певних умов ці складові можуть бути частково або повністю вилучені з інформації). Одним із прикладів такої інформації є звукова інформація, призначена для прослуховування. Людський слух, як відомо, сприймає досить обмежений спектр частот, тому зменшення числа збережених звукових гармонік не завжди приводить до погіршення якості звуку.

Принципово інший приклад використання методів економного кодування із втратами – передача телеграфних повідомлень. З метою збільшення швидкості передачі інформації з повідомлень видаляються слова (розділові знаки й т.д.), які не несуть значимого значеннєвого навантаження.

Дана лекція **присвячена** викладу одного з методів першої групи – методів **економного кодування інформації без втрат**. Якщо методи економного кодування із втратами можуть бути орієнтовані на довільний тип інформації, методи економного кодування без втрат використовуються винятково для обробки дискретної інформації.

Уявимо собі деяку гіпотетичну систему кодування, у якій різним символам відповідають коди різної довжини. У методі обмеженого алфавіту довжина коду для всіх символів була однаковою, тобто ми мали справу з так званим рівномірним кодом. Нова система кодування повинна містити в собі зовсім інший тип кодів – **коди змінної довжини**.

У чому ж складається **перевага** нової системи кодування? Виявляється, що використання кодів змінної довжини дозволяє врахувати статистичні особливості появи символів в інформаційному повідомленні. Символам, що зустрічаються часто, доцільно ставити у відповідність короткі коди, а символам, що зустрічаються рідше, – довгі коди. Отримане в такий спосіб інформаційне подання може виявитися істотно більш ефективним у порівнянні з поданням, отриманим із застосуванням рівномірного коду.

Отже, припустимо, що існує спосіб кодування, що дозволяє використовувати кодові комбінації змінної довжини. Задамося питанням: як повинна виглядати залежність довжини коду від імовірності появи символу, що кодується, в інформаційному повідомленні? З теорії, основи якої були закладені К. Шенноном у роботі, слідує, що в системі подання інформації з основою m (при $m = 2$ це двійкова або бінарна система) символу, ймовірність появи якого дорівнює p, оптимально й, що особливо важливо, теоретично припустимо ставити у відповідність код довжини $-\log_m p$. Це означає, наприклад, що символи 128-символьного набору ASCII, що з'являються в потоці інформації з рівною ймовірністю, у двійковій системі подання доцільно кодувати $-\log_2 1/128 = 7$ бітами.

Наведена вище формула лежить в основі теорії економного кодування. Її наслідком є раніше висловлене положення про те, що ефективність подання інформації цілком і повністю залежить від властивостей самої інформації. Для кількісної оцінки цих властивостей Шеннон вводить дві тісно зв'язані між собою характеристики: ентропія (*entropy*) і надмірність (*redundancy*).

Відповідно до цього, оптимальна довжина коду символу визначається ймовірністю його появи на виході джерела інформації. У реальних задачах ми звичайно не знаємо цю ймовірність, однак можемо одержати її оцінку на основі деякої інформаційної моделі.

У даній лекції нас не будуть цікавити способи обчислення імовірнісних оцінок; ми займемося вивченням методик генерації систем кодів змінної довжини по вже заданих імовірнісних розподілах. Такі системи звичайно одержують або з використанням префіксного кодування, або з використанням арифметичного кодування.

Префіксне кодування є найпростішим методом генерації кодів змінної довжини. Коди, одержувані з використанням префіксного кодування, мають цілу довжину в одиницях подання інформації. Перевага префіксного кодування заключається в його простоті й високій продуктивності, а недолік – у неможливості створювати коди нецілої довжини, що, природно, негативно позначається на ефективності кодування. По зрозумілих причинах (Мінімальна довжина префіксного коду дорівнює інформаційній одиниці), найбільші втрати в ефективності при використанні префіксного кодування спостерігаються у випадку частої генерації кодів, теоретично оптимальна довжина яких близька до нуля.

На відміну від префіксного кодування, арифметичне кодування дозволяє генерувати коди як цілої, так і нецілої довжини. Будучи теоретично оптимальним

методом, арифметичне кодування перевершує в ефективності префіксне кодування. Воно також випереджає префіксне кодування у швидкості побудови системи кодів, однак через підвищену складність нерідко помітно уступає у швидкості самого кодування (мається на увазі процес генерації кодової послідовності).

Незважаючи на деякі істотні переваги арифметичного кодування, даний метод до останнього часу був недостатньо розповсюджений на практиці. На сьогоднішній день у більшості комерційних додатків для побудови системи кодів змінної довжини використовується кодування Хаффмана, що є кращим з погляду ефективності, реалізацією префіксного кодування. Арифметичне кодування застосовується лише в тих випадках, коли потрібно домогтися максимально можливої якості інформаційного подання й коли швидкість роботи не має вирішального значення.

Префіксне кодування

Префіксне кодування засноване на виробленні систем кодів змінної довжини, які прийнято називати системами префіксних кодів (або кодів префікса). Особливість цих систем полягає в тому, що в межах кожної з них більше короткі по довжині коди не збігаються з початком (префіксом) більше довгих кодів – так звана властивість префікса.

Розглянемо систему із трьох двійкових кодів: {«0», «10», «11»}. Як видно, жоден із цих кодів не є початком іншого. Таким чином, про дану систему можна говорити як про систему префіксних кодів. Нехай трьом обраним кодам відповідають символи «А», «П» і «Ш». Тоді інформаційному повідомленню «АПАШ» буде відповідати кодова послідовність «010011». Спробуємо відновити вихідне повідомлення по даній кодовій послідовності. Перший нульовий біт можна інтерпретувати тільки як символ «А», тому що коди, що відповідають символам «П» і «Ш», починаються з одиниці. За першим бітом слідує одиничний біт, що може бути початком кодів відразу двох символів: «П» і «Ш». Визначити наступний символ дозволяє значення третього біта, що однозначно вказує на символ «b». На даному етапі відновлена частина повідомлення здобуває вид «АП». Слідом за кодом символу «П» знову треба нульовий біт. Як ми з'ясували, це код, що відповідає символу «А». Завершують кодову послідовність два одиничних біти. Так само, як і в розглянутому вище випадку, значення останнього з них дозволяє однозначно відтворити останній символ закодованого повідомлення – «Ш». Отже, нам удалося правильно декодувати всю кодову послідовність. Як неважко зрозуміти, це стало можливим завдяки виконанню властивості префікса.

Що дає на практиці така система префіксних кодів? У наведеному прикладі символ «a» зустрічається в повідомленні «abac» з відносною частотою 1/2, а символи «b» і «c» – з відносною частотою 1/4. Припустимо, що інформаційне джерело, що породило дане повідомлення, є джерелом без пам'яті. Через брак інших відомостей про джерело, логічно вважати ймовірності появи символів його на виході рівними відносним частотам. Тоді оптимальна довжина коду для символу «a» повинна бути дорівнює $-\log_2 1/2 = 1$ біт, а для символів «b» і «c»

$-\log_2 1/4 = 2$ біта. Як видно, обрана система префіксних кодів перебуває в повній відповідності з передбачуваним імовірнісним розподілом.

Застосування префіксного кодування в розглянутому прикладі дозволило одержати оптимальне з погляду компактності подання інформаційного повідомлення. Однак, якщо до даного повідомлення додати ще один символ «a», використання префіксного кодування вже не дасть оптимального результату. Символ «a» буде зустрічатися в новому повідомленні з відносною частотою 3/5, а символи «b» та «c» – з відносною частотою 1/5. Оптимальні довжини кодів для символів «a», «b» та «c» будуть відповідно рівні $-\log_2 3/5 \approx 0{,}74$, $-\log_2 1/5 \approx 2{,}35$ та $-\log_2 1/5 \approx 2{,}35$ біта. Таким чином, для подання нового повідомлення досить приблизно 6.86 біта. Для порівняння, максимально можлива ефективність префіксного кодування в цьому випадку становить 7 біт.

Системи префіксних кодів звичайно одержують побудовою кодових дерев. Ступінь розгалуження в цих деревах залежить від основі системи подання інформації. Розглянемо випадок двійкової системи подання, якій відповідають бінарні кодові дерева (наступне міркування легко поширюється й на випадок m-*ічної* системи подання). Як відомо, кожний внутрішній вузол бінарного дерева має до двох вихідних ребер. Пронумеруємо ці ребра так, щоб одному з них відповідав двійковий символ «0», а іншому – «1». Так як в дереві не може бути циклів, від кореневого вузла до листового вузла завжди можна прокласти єдиний маршрут. Якщо ребра дерева пронумеровані, то кожному такому маршруту відповідає деяка унікальна двійкова послідовність. Множина всіх таких послідовностей, утворить систему префіксних кодів.

Легко зрозуміти, що будь-якій системі префіксних кодів однозначно відповідає деяке кодове дерево (наприклад, розглянутій системі префіксних кодів {«0», «10», «11»} відповідає кодове дерево, зображене на мал.1). Таким чином, виконання властивості префікса можна інтерпретувати як наявність особливої деревоподібної організації усередині системи кодів змінної довжини. Крім такого геометричного трактування властивості префікса, можлива також і алгебраїчне трактування, що говорить, що система префіксних кодів з натуральними довжинами $l_1 ... l_N$ в системі подання інформації з підставою m може існувати тоді й тільки тоді, коли виконується нерівність Крафта:

$$\sum_{i=1}^{N} m^{-l_i} \le 1.$$

Для того щоб переконатися в тотожності цих, на перший погляд несхожих тверджень, досить помітити, що виконання нерівності Крафта фактично еквівалентно існуванню дерева з N листовими вузлами, що перебувають на відстанях l_i від кореневого вузла.

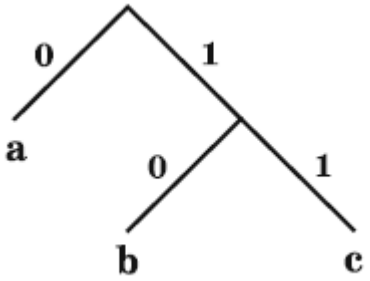

Рисунок 8.1 – Бінарне кодове дерево для системи префіксних кодів {«0», «10», «11»}(коди відповідають символам «a», «b» і «c»)

Кодові дерева необхідно будувати таким чином, щоб довжини маршрутів, що ведуть від кореневого вузла до листових вузлів, були максимально наближені до теоретично оптимальних довжин кодів, що відповідають різним символам інформаційного алфавіту. Помітимо, що мова тут йде лише про довжини маршрутів, але не про самі маршрути. Справа в тому, що нумерація ребер кодового дерева може бути здійснена абсолютно довільним образом, що обумовлює довільність у виборі маршруту, тобто, в остаточному підсумку, довільність у виборі складу префіксного коду. Інакше кажучи, конкретна комбінація нулів і одиниць, яка складає префіксний код, не грає особливої ролі, а визначальне значення має тільки його довжина. Таким чином, для заданого імовірнісного розподілу може бути побудоване відразу кілька кодових дерев, що відповідають різним оптимальним системам префіксних кодів.

Отже, як же будувати ці дерева? Найбільш відомими алгоритмами побудови кодових дерев з конфігурацією, що відповідає заданому ймовірнісному розподілу, є алгоритми Шеннона-Фано й Хаффмана. Як було зазначено вище, система кодів, отримана із застосуванням алгоритму Хаффмана, є кращою з погляду якості інформаційного подання серед всіх можливих систем префіксних кодів. Алгоритм Шеннона-Фано менш ефективний, однак він істотно більше простий і наочний. Крім алгоритмів Шеннона-Фано й Хаффмана, можна також виділити алгоритми Голомба й Райса. Розглянемо найпоширеніші алгоритми префіксного кодування стосовно до двійкової системи подання інформації.

Алгоритм Шеннона-Фано

Розглянутий алгоритм запропонували незалежно друг від друга Р. Фано і К. Шеннон. Алгоритм являє собою рекурсивну процедуру, що дозволяє на основі заданого імовірнісного розподілу появи символів на виході джерела інформації поетапно формувати систему, що відповідає йому, префіксних кодів.

Спочатку кожному символу інформаційного алфавіту ставиться у відповідність код нульової довжини («порожній» код) і вага, рівна ймовірності появи символу на виході інформаційного джерела в рамках обраної інформаційної моделі. Всі символи алфавіту сортуються по убуванню або зростанню їхніх ваг, після чого впорядкований ряд символів у деякому місці ділиться на дві частини так, щоб у кожній з них сума ваг символів була приблизно однакова. До коду символів, що належать однієї із частин, додається

«0», а до коду символів, що належать іншої частини, додається «1» (додається значення, що, формує черговий крайній правий розряд коду). Як неважко зрозуміти, кожна із зазначених частин сама по собі є впорядкованим рядом символів. Кожний із цих рядів, якщо він містить більше одного символу, у свою чергу ділиться на дві частини відповідно до описаного вище принципом, і до коду символів знову додаються відповідні двійкові значення й т.д. Процес завершується тоді, коли у всіх отриманих у такий спосіб рядах залишається рівно по одному символу.

Як видно, алгоритм Шеннона-Фано в дійсності не будує кодове дерево, однак його роботу можна проілюструвати такою побудовою. При поділі групи символів відбувається як би розгалуження деякого вузла бінарного дерева (листовий вузол стає вузлом батьком для двох новостворених дочірніх вузлів), а присвоєння нулів і одиниць рівносильно встановленню відповідності між кодами й маршрутами руху по дереву від кореневого вузла до листового вузла. Робота алгоритму Шеннона-Фано проілюстрована на наступному прикладі (рис. 8.2).

Як же ми ділили на групи ? Досить просто:

Розміщаємо символи по ймовірності появи.

1. ймовірність першої групи (**p1**) і другої (**p2**) дорівнює нулю;
2. **p1** <= **p2** ?
   - так: додати в першу групу символ з початку таблиці;
   - ні: додати в другу групу символ з кінця таблиці;
3. якщо всі символи розділені на групи, то завершити алгоритм, інакше перейти до кроку **2**

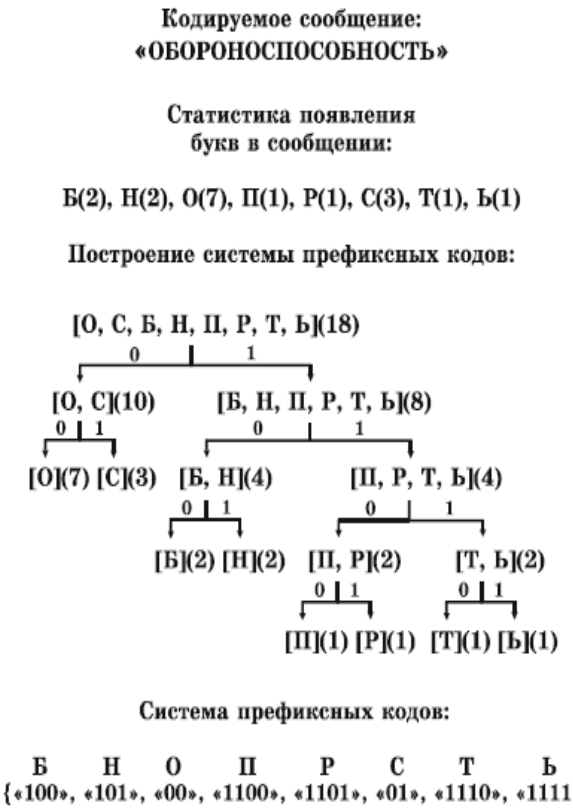

Рисунок 8.2 – Ілюстрація роботи алгоритму Шеннона-Фано на прикладі кириллиці

У цьому випадку використовується інформаційна модель, у якій інформаційне джерело, що породжує, розглядається як джерело без пам'яті. Імовірності появи символів у даній моделі заміняються частотами їхнього входження в інформаційне повідомлення. Надалі така модель буде йменуватися нульовою моделлю.

Наскільки обґрунтований такий спосіб побудови системи кодів змінної довжини? Як виявляється, він не універсальний навіть у рамках префіксного кодування. Для доказу досить знайти оптимальну систему префіксних кодів для 5-ти символів, імовірності появи яких рівні 5/13, 2/13, 2/13, 2/13 і 2/13, і переконатися в тому, що система, отримана з використанням алгоритму Шеннона-Фано, у цьому випадку виявляється менш вигідною.

Проте алгоритм Шеннона-Фано досить ефективний. Нескладно показати, що довжина коду символу з імовірністю появи *p*, отриманого з використанням даного алгоритму, перевищує теоретично оптимальне значення $-\log_2 p$ менш чим на 1 біт. Ефективність алгоритму знаходить підтвердження й на практиці: як правило, алгоритм Шеннона-Фано дуже незначно уступає в ефективності оптимальному алгоритму префіксного кодування – алгоритму Хаффмана.

Алгоритм Хаффмана

Хаффман запропонував будувати кодове дерево не зверху до низу, як це неявно робиться в алгоритмі Шеннона-Фано, – від кореневого вузла до листових вузлів, а знизу нагору – від листових вузлів до кореневого вузла. Такий підхід виявився найбільш ефективним і одержав досить широке поширення.

На початковому етапі роботи алгоритму кожному символу інформаційного алфавіту ставиться у відповідність вага, рівна ймовірності (частоті) появи даного символу в інформації. Символи містяться в список, що сортується по убуванню ваг. На кожному кроці (ітерації) два останніх елементи списку поєднуються в новий елемент, що потім міститься в список замість двох поєднуваних елементів. Новому елементу списку ставиться у відповідність вага, дорівнює сумі ваг елементів, що заміщаються. Кожна ітерація закінчується впорядкуванням отриманого нового списку, що завжди містить на один елемент менше, ніж старий список. Паралельно з роботою зазначеної процедури здійснюється послідовна побудова кодового дерева. На кожному кроці алгоритму будь-якому елементу списку відповідає кореневий вузол бінарного дерева, що складається з вершин, які відповідають елементам, об'єднанням яких був отриманий даний елемент. При об'єднанні двох елементів списку відбувається об'єднання відповідних дерев в одне нове бінарне дерево, у якому кореневий вузол відповідає новому елементу, що поміщається в список, а елементам списку, що заміщуються, відповідають дочірні вузли цього кореневого вузла. Алгоритм завершує роботу, коли в списку залишається один елемент, що відповідає кореневому вузлу побудованого бінарного дерева. Це дерево називається деревом Хаффмана. Система префіксних кодів може бути отримана шляхом присвоювання конкретних двійкових значень ребрам цього дерева. Приклад побудови дерева Хаффмана наведений на рис. 4.3.

Отже, система кодів, отримана в результаті побудови дерева Хаффмана, є оптимальною системою префіксних кодів. Звідси, однак, у жодному разі не треба висновок про те, що префіксні системи, отримані з використанням інших алгоритмів, завжди менш ефективні. Так, на розглянутих прикладах (див. рис. 8.2 та 8.3) добре видно, що алгоритми Шеннона-Фано й Хаффмана, хоча й будують зовсім різні кодові системи (коди для тих самих символів розрізняються як по довжині, так і по складу), у даному конкретному випадку виявляються однаково ефективними – при використанні кожного із цих алгоритмів об'єм подання повідомлення «ОБОРОНОСПОСОБНОСТЬ» дорівнює 48 бітам. Таким чином, алгоритм Хаффмана є не єдиним алгоритмом, що дозволяє будувати оптимальні системи префиксних кодів, однак єдиним до кінця універсальним алгоритмом – на відміну від інших алгоритмів, даний алгоритм завжди оптимальний.

Основний **недолік** алгоритму Хаффмана полягає в складності процесу побудови системи кодів. Багато альтернативних алгоритмів префіксного кодування, зокрема алгоритм Шеннона-Фано, представляються щодо цього більш кращими. Справедливості заради помітимо, що дана особливість рідко враховується при виборі алгоритму кодування, тому що, при необхідності частих перебудувань системи кодів, як правило, застосовуються спеціалізовані підходи.

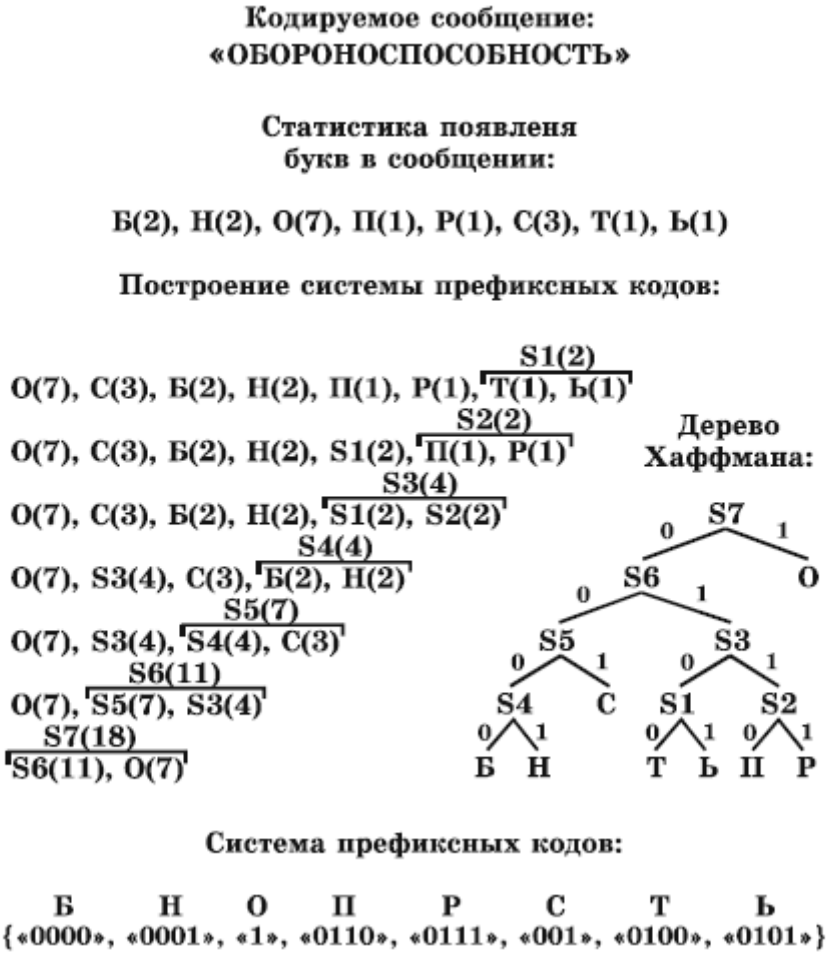

Рисунок 8.3 – Ілюстрація роботи алгоритму Хаффмана на прикладі кирилиці

Алгоритм Хаффмана є найпоширенішим алгоритмом генерації коду змінної довжини. Протягом багатьох років кодування Хаффмана, засноване на найпростішій нульовій інформаційній моделі, виступало в ролі самостійного алгоритму економного кодування.

Сьогодні алгоритм Хаффмана використовується винятково як спосіб одержання систем кодів змінної довжини. Він знаходить своє втілення в більшості утиліт стиску й архівації інформації.

Основні методи економного кодування без втрат послідовної дискретної інформації

Методи цієї групи підрозділяються в такий спосіб:

1. Статистичні методи.

1.1. Метод PPM

1.2. Метод CTW

1.3. Метод DMC

1.4. Економне кодування, засноване на використанні штучних нейронних мереж

2. Словникові методи
2.1. Алгоритми сімейства LZ77
2.2. Алгоритми сімейства LZ78
2.3. Алгоритми групи LZRW
2.4. Комбіновані алгоритми сімейства LZ77
2.5. Алгоритм LZP
3. Контекстні методи
3.1. Асоціативне кодування (метод ACB)
3.2. Метод Барроуза-Уілера

Методи перших двох груп будуються на основі відповідно статистичних і словникових моделей. Вони досить численні й представляють майже всі існуючі підходи в області економного кодування дискретної інформації. Методи третьої групи менш численні й погано піддаються класифікації по типу інформаційної моделі. На сьогоднішній день контекстна група містить у собі тільки два методи, в одному з яких використовується модель, яку варто скоріше віднести до словникових моделей, а в іншому – модель особливого типу, що не є ні словниковою, ні статистичною. Причиною виділення даних методів в окрему групу послужило те, що вони оперують інформаційними об'єктами – контекстами даних. Розглянемо докладніше основні методи кожної з перерахованих груп.

Статистичні методи

Для здійснення ефективного економного кодування повідомлень, що надходять із виходу джерела інформації, потрібне знання його характеристик. При розгляді джерел з пам'яттю, як ми знаємо, за такі характеристики прийнято брати умовні ймовірності появи символів у повідомленні слідом за різними символьними послідовностями. У випадку, коли немає ніякої додаткової інформації про джерело, визначити ці ймовірності можна тільки шляхом статистичного аналізу його інформаційної вибірки. Даний **принцип** лежить в основі статистичних методів економного кодування. Послідовність символів, які безпосередньо передують деякому символу в інформаційному повідомленні, прийнято називати **контекстом цього символу**, а довжину цієї послідовності – **порядком контексту**. На основі статистики, що збирається під час обробки інформації, статистичний метод дозволяє оцінити ймовірність появи в поточному контексті довільного символу або послідовності символів. Оцінка ймовірності визначає довжину коду, що, як правило, генерується з використанням арифметичного кодування. Статистичні методи здебільшого є адаптивними методами. Обчислення імовірнісних оцінок здійснюється тим самим способом на етапах кодування й декодування, що дозволяє не зберігати опис інформаційної моделі. Статистичні методи відрізняються способом одержання оцінок умовних ймовірностей появи символів у різних контекстах. На сьогоднішній день можна виділити чотири основних способи одержання таких оцінок, які лягли в основу відповідно чотирьох статистичних методів: методу PPM, методу DMC, методу CTW і методу, заснованого на використанні

нейронних мереж. Найбільше поширення серед перерахованих методів одержав запропонований ще в середині 80-х років метод PPM, довгий час практично що безроздільно був найбільш ефективним методом економного кодування. Метод DMC, хоча й з'явився практично одночасно з методом PPM, все-таки не одержав такого широкого поширення. Він має меншу ефективність у порівнянні з методом PPM і становить інтерес в основному тільки з ідейної точки зору. Два методи, що залишилися відносяться до сучасних статистичних методів економного кодування. Метод CTW став результатом цілеспрямованих теоретичних досліджень по пошуку так званого «універсального» методу економного кодування. На відміну від методу CTW, метод економного кодування, заснований на використанні нейронних мереж, спочатку взагалі не мав прямого відношення до області економного кодування. Він розроблявся як загальний метод моделювання й пророкування в рамках зовсім іншої дисципліни – теорії штучних нейронних мереж. Обидва зазначених методів відбивають найбільш перспективні напрямки надалі розвитку статистичного підходу в економному кодуванні й уже сьогодні можуть скласти конкуренцію методу PPM.

Словникові методи

У попередньому викладі інформаційні закономірності розглядалися як чисто імовірнісні. Альтернативне трактування має на увазі використання так званого комбінаторного підходу. У новому трактуванні проявом інформаційних закономірностей є присутність в інформації одних символів або символьних комбінацій і повна відсутність інших. Помітимо, що в цьому випадку ми не говоримо про імовірнісні міжсимвольні взаємозв'язки, а просто констатуємо факт наявності або відсутності окремих символьних ланцюжків в інформації.

Кодування в рамках комбінаторного підходу ґрунтується на побудові кодових систем, що містять коди тільки тих інформаційних послідовностей, які реально породжуються інформаційним джерелом. Методи, що представляють у явному виді зазначений підхід, – методи словникової групи, уперше описані в роботах А. Лемпела і Я. Зіва

Словникові методи використовують моделі, основу яких становить інформаційна структура, іменована словником. Під час роботи словникового методу словник містить у собі частини вже обробленої інформації, що виступають як матеріал, на основі якого здійснюється кодування. У процесі кодування складові символьної послідовності, що надходить із виходу кодуємого джерела, кодуються за допомогою посилань на ідентичні їм елементи словника (збіги). Словникові методи відрізняються друг від друга способом організації словника, схемою пошуку збігів і видом посилання на знайдений збіг.

Розбивка оброблюваної послідовності на кодуємі складові являє собою важкорозв'язну задачу, від оптимальності рішення якої у високому ступені залежить ефективність словникового методу.

Розглянемо цю проблему на конкретному прикладі. Припустимо, що необхідно обробити послідовність символів «АПАШ», і при цьому відомо, що словник містить чотири рядки: «А», «АП», «Ш», «ПАШ». Очевидно, що

послідовність можна розбити на складовими двома різними способами: «АПАШ» = «АП» + «А» + «Ш» або «АПАШ» = «А» + «ПАШ». В основі першого способу, називаного «жадібним» розбором (*greedy parsing*), лежить вибір найбільш довгого збігу при послідовній розбивці послідовності ліворуч праворуч. Другий спосіб, іменований оптимальним розбором (*optimal parsing*), полягає в знаходженні найкращої розбивки із всіх можливих. Через колосальну обчислювальну складність оптимальний розбір не використовується на практиці. Замість нього в більшості випадків використовується «жадібний» розбір, що легко реалізуємо й, крім того, зручний при послідовній обробці інформації. Однак «жадібний» розбір не оптимальний з погляду ефективності. Для її підвищення рекомендується робити пошук збігів, починаючи не тільки з першої оброблюваної позиції, але й декількох наступних позицій. Збіг варто вибирати, керуючись міркуваннями оптимальності довжини одержуваної кодової комбінації. Даний підхід прийнятий називати ледачим пошуком або ледачим зіставленням (*lazy matching*).

Контекстні методи

Сутність марківського ланцюга як процесу породження інформації полягає в наявності взаємозв'язку між повідомленням і його передісторією, тобто, дотриманням строгої термінології, між повідомленням і його контекстом. Однак, якщо повідомлення залежать від контекстів, у яких вони з'являються, можна стверджувати, що вони також залежать і від наступних за ними повідомлень. У зв'язку із цим доцільно трохи розширити поняття контексту й називати контекстом повідомлення не тільки те, що безпосередньо передує йому, але також і те, що безпосередньо йде за ним. Для проведення розходження між цими, загалом кажучи, неідентичними об'єктами назвемо перший з них **лівобічним контекстом**, а другий – **правобічним контекстом**.

Головне, що відрізняє два даних типи контекстів – це хронологічний порядок їхнього проходження: лівосторонній контекст деякого повідомлення завжди передує його правобічному контексту.

Розглянемо два методи економного кодування, в основу яких був покладений облік залежності між символьним контекстом і його доповненням. Незважаючи на ідейну спільність, ми маємо справу із зовсім різними підходами, в одному з яких генерація оброблюваної інформації розглядається як послідовний процес, а в іншому – як одноразовий акт. Метод АСВ, що представляє перший підхід, призначений для послідовної обробки інформації, що надходить на вхід кодера відповідно до умовної хронології її народження. Так само, як і більшість розглянутих методів, метод АСВ ставиться до потокових методів економного кодування. Представником другого, менш стандартного підходу є метод Барроуза-Уілера. Особливість даного методу – непослідовна обробка інформації, що можлива тільки при наявності на будь-якому етапі роботи всього об'єму оброблюваної інформації. Зазначені методи є сучасними високоефективними методами економного кодування, уже сьогодні застосовуваними в реальних додатках і маючими непогані перспективі для подальшого практичного використання.

8.2. Принципи завадостійкого кодування. Основні характеристики завадостійких кодів. Класифікація завадостійких кодів. Математичний опис процесу кодування і декодування

Принципи завадостійкого кодування. Основні характеристики завадостійких кодів

Розглянемо загальну постановку задачі завадостійкого кодування повідомлень.

Від джерела інформації надходить послідовність елементарних повідомлень x_i. Кодування полягає в тому, що послідовність символів джерела заміняється послідовністю m-ічних кодових символів (надалі будемо розглядати тільки двійкові коди, для яких m=2). Зазначене перетворення є взаємнооднозначним, що й дозволяє здійснити в приймачі декодування, тобто відновити повідомлення по прийнятій кодовій комбінації.

У результаті дії перешкод деякі кодові символи в приймачі виділяються перекрученими – одиничні елементи комбінації сприймаються як нульові й навпаки.

Може бути передбачений і режим стирання символів, коли рішення про прийом деяких з них не може бути виконане з необхідною надійністю (упевненістю).

Завадостійке кодування повідомлень має своєю метою перетворення повідомлень x_i, i=1,…,m у послідовність кодових символів x_j, j=1, …, n, що володіє властивістю або виявлення присутності в ній помилок, або виявлення й виправлення цих помилок.

Відразу відзначимо, що не існує кодів кінцевої довжини, що дозволяють виявити й, тим більше, виправити всі можливі помилки. Можна побудувати лише коди, що виявляють або виправляють деяке число помилок певного виду (звичайно найбільш ймовірних або найнебезпечніших).

Визначена мета може бути досягнута тільки введенням надмірності, тобто збільшення числа кодових символів у переданій послідовності стосовно символів, що забезпечують однозначне подання переданих повідомлень (інформаційним символам). Наявність додаткових (надлишкових) символів дозволяє накласти на передані послідовності символів (кодові комбінації) додаткові зв'язки (умови) між ними, перевірка яких на прийомній стороні дає можливість виявити й виправити помилки. Вся сукупність можливих кодових комбінацій у цьому випадку називається завадостійким (коригувальним або надлишковим) кодом.

При використанні завадостійкого коду передаються в канал не всі кодові комбінації, що можна сформувати з наявного числа розрядів, а лише що мають певну властивість і що **називаються дозволеними**. Інші комбінації, що не використовуються, називаються **забороненими**. Введення додаткових розпізнавальних ознак в комбінації, що передаються дозволяє істотно підвищити правильність класифікації.

Завадостійкі коди поділяються на:

– коди, що **виявляють** помилки;

– коди, що **виправляють** помилки.

При використанні кодів, що виявляють помилки вся численність n-розрядних комбінацій розбивається на дві підмножини, що не перехрещуються. Одна підчисленність називається **дозволеною**, а інша – **забороненою**. Передаються тільки дозволені кодові комбінації, що мають певною властивістю. Якщо прийнята кодова комбінація відноситься до дозволених, то вважається, що помилки немає. На прийомній стороні відомо, які з комбінацій є дозволеними, а які забороненими. Тому, якщо передана комбінація в результаті помилки перетвориться в деяку заборонену комбінацію, то така помилка буде виявлена, а за певних умов і виправлена. При побудові кодів, що виправляють помилки вся численність кодових комбінацій розбивається на ряд підмножин, що не перехрещуються. В кожній з підмножин одна дозволена комбінація. При прийомі будь-якої комбінації з даної підмножин споживачу видається дозволена комбінація цієї підмножин.

Можливості по виявленню або виправленню помилок визначаються числом позицій, на яких відрізняються дозволені кодові комбінації, т. т. **кодовою відстанню**.

Кодова відстань між i-ю і j-ю кодовими комбінаціями (**відстань за Хеммінгом**) визначається за формулою:

$$d_{ij} = \sum_{k=1}^{n} (x_{ik} \oplus x_{jk}),$$

де x_{ik} і x_{jk} – значення символів k – й позиції i – й і j – й кодових комбінацій. Число відмінних від 0 символів у кодовому слові x_i називається його **вагою** w_i.

Наприклад, між кодовими комбінаціями 1100101 і 1011100, що відрізняються символами в чотирьох розрядах l=(0111001), відстань дорівнює чотирьом.

Для будь-якого коду $1 \le d \le n$. Мінімальна відстань між дозволеними комбінаціями в даному коді називається **мінімальною кодовою відстанню** й позначається d_{min}.

Таким чином, якщо код має кодову відстань d_{min}, то це позначає, що дозволені комбінації цього коду відрізняються друг від друга не менш чим в d_{min} символах і, отже, тільки поява d_{min} і більше помилок у прийнятій комбінації можуть перевести її в іншу дозволену комбінацію.

З іншого боку, якщо число помилок у комбінації буде менше ніж d_{min}, то стає зрозумілим, що така комбінація помилок буде неодмінно приводити до забороненої комбінації.

В загальному випадку необхідна кодова відстань для забезпечення **виявлення всіх помилок кратності до** t_0 **включно** визначається вираженням $t_0 = d + 1$.

При **виправленні помилок кратності до** t_u **включно** кодова відстань повинна бути рівна $d = 2t_u + 1$. З цих виразів видно, що $t_0 = 2t_u$.

Необхідна кодова відстань при **виправленні помилок кратності до** t_u **включно і виявлення помилок кратності від** $t_u + 1$ **до** t_0 **повинна** бути рівна $d \geq t_u + t_0 + 1$. Необхідна кодова відстань, а отже, і завадостійкість коду визначається **надмірністю коду**, т. є. числом введених перевірочних символів.

Визначимо кількість перевірочних розрядів, необхідну для виправлення t_u **помилок**. Для цього необхідно, щоб за допомогою перевірочних розрядів можна було описати наступні ситуації:

– помилка буде відсутня – 1-й випадок;

– одинока помилка – C_n^1 випадків;

– двократна помилка – C_n^2 випадків;

....

– помилка кратності t_u – $C_n^{t_u}$ випадків,

де C_n^i – число сполучень (комбінацій) з n по i. Таким чином, кількість перевірочних розрядів для виправлення помилок кратності t_u і менш визначається з наступної нерівності:

$$2^k \geq \sum_{i=0}^{t_u} C_n^i, \Rightarrow k \geq \log_2 \sum_{i=0}^{t_u} C_n^i.$$

Даним вираженням можна скористуватися і для знаходження числа перевірочних розрядів для виявлення помилок. Для цього необхідно використати той факт, що число помилок, що виявляються в два рази більше числа помилок, що виправляються.

Отже, для виявлення t_0 помилок кількість перевірочних одиничних елементів повинно задовольняти нерівності:

$$k \geq \log_2 \sum_{i=0}^{t_o/2} C_n^i.$$

Важливими показниками ефективності коду є **коефіцієнти помилок**, що виявляються $K_{ВП}$ та не виявляються $K_{НВ}$, під якими розуміють відношення числа кодових комбінацій з виявленими (невиявленими) помилками до числа всіх можливих комбінацій. Помилки не виявляються, коли кодова комбінація, що передається під впливом завад перетворюється в іншу дозволену. Число всіх дозволених комбінацій при m інформаційних розрядах рівно 2^m. Отже, при передачі будь якої кодової комбінації можливе число випадків невиявлених помилок буде рівно $N_{НП} = 2^m\text{-}1$. Оскільки число всіх можливих помилкових комбінацій рівно 2^n, де n – кількість розрядів у кодовій комбінації, то вираження для визначення величини $K_{НП}$ буде мати вигляд:

$$K_{НП} = \frac{2^m - 1}{2^n} = \frac{1}{2^{n-m}} - \frac{1}{2^n} = \frac{1}{2^k} - \frac{1}{2^n},$$

де $k = n - m$ – число надлишкових розрядів. Так як число всіх заборонених кодових комбінацій рівно $2^n - 2^m$, то коефіцієнт може бути визначений з вираження:

$$K_{вп} = \frac{2^n - 2^m}{2^n} = 1 - \frac{1}{2^k}.$$

Класифікація завадостійких кодів

Коди поділяються (рис. 4.4) на:

– блочні (блокові);

– безперервні.

До **блочних** відносяться коди, у яких кожному повідомленню ставиться в однозначну відповідність блок з n символів. **Безперервні** коди представляють безперервну послідовність інформаційних і перевірочних розрядів.

Блочні коди діляться на:

– рівномірні;

– нерівномірні.

Рівномірні коди діляться на:

– систематичні;

– несистематичні.

Під **систематичними** розуміють код, в якому розряди можуть бути поділені на перевірочні і інформаційні. При цьому їхні місця в кодовій комбінації цілком визначені. Несистематичні коди цю властивість не мають.

Окрім цього, коди діляться на:

– лінійні;

– нелінійні.

Лінійні коди це такі, у яких сума за модулем 2 дозволених комбінацій дасть дозволену комбінацію того же коду. Нелінійні коди означену властивість не мають. Передані повідомлення визначаються m інформаційними символами. Надлишкові (контрольні, перевірочні) k символів доповнюють кодову комбінацію до $n = m + k$ символів, у силу чого такі блокові коди позначають як (n,m) коди

Більшість кодів, що застосовуються на практиці, відноситься до лінійних. Найбільше поширення серед блокових кодів знайшли коди з перевіркою на парність, з повторенням символів, рівноважні коди, коди Хеммінга, коди Ріда-Малера й, найбільш великий клас кодів – циклічні коди. До останнього класу належать коди Боуза-Чоудхури-Хоквингема (БЧХ) і коди Ріда-Соломона (РС), що одержали найбільше поширення в техніці військового зв'язку.

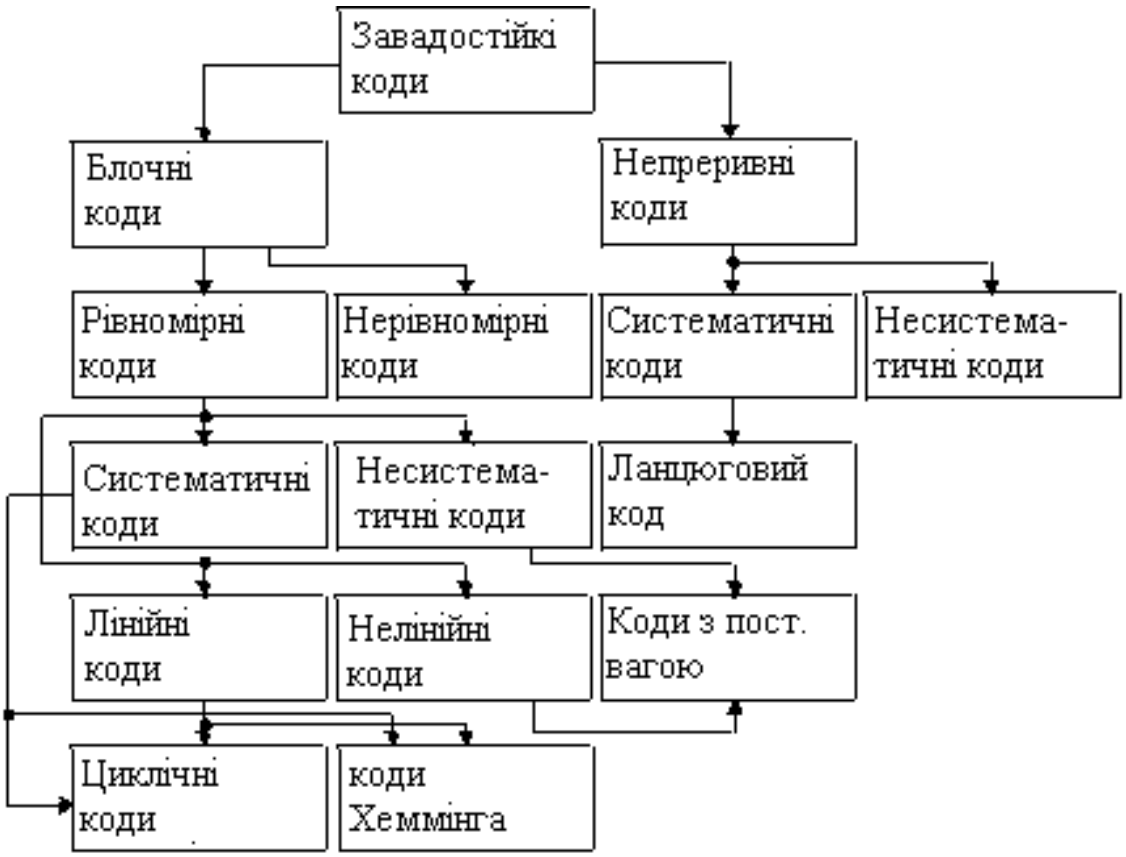

Рисунок 8.4 – Класифікація завадостійких кодів

Математичний опис процесу кодування і декодування

При завданні коду звичайно вказують, які інформаційні елементи беруть участь в формуванні кожного з k перевірочних розрядів. Наприклад, для коду з n = 5, m =3, k = 2 кожний перевірочний розряд $П_i$ визначається складанням за модулем 2 по правилу:

$$П_1 = I_1 \oplus I_2;\ П_2 = I_2 \oplus I_3,$$

де I_i – інформаційні розряди.

Комбінація такого коду записується в вигляді $П_2, П_1, I_3, I_2, I_1$. При завданні коду можна вказати всі дозволені для цього коду комбінації. Для лінійних кодів засіб завдання можна значно спростити. Для m інформаційних розрядів число всіх дозволених кодових комбінацій буде рівно 2^m. Виберемо з усіх кодових комбінацій тільки лінійно незалежні. Під **лінійно незалежними кодовими комбінаціями** розуміють такі, сума за модулем 2 яких (в будь-якому поєднанні) не рівна нулю. Для наведеного прикладу такими комбінаціями можуть бути комбінації вигляду:

1) 01001	2) 01001	3) 01001
11010	10011	10011
10100	01110	00111

Всі інші кодові комбінації можна отримати складанням за модулем 2 лінійних незалежних комбінацій. Звичайно лінійно незалежні кодові комбінації записують в вигляді матриці розміром $n \times m$, яку називають **породжуючою матрицею** і позначають $G(n \times m)$. Найбільш часто породжуючі матриці записують так званій **канонічній формі**. При цьому перші або останні m стовпчиків утворять одиничну матрицю. Інші стовпчики вказують правила формування перевірочних розрядів. Так, для наведеного прикладу породжуюча матриця в канонічній формі має вигляд:

$$G(5,3) = \left\| \begin{matrix} 10100 \\ 11010 \\ 01001 \end{matrix} \right\|.$$

З прикладу видно, що останні три стовпчика складають одиничну матрицю; другий стовпчик вказує, що при формуванні першого перевірочного розряду беруть участь перший і другій інформаційні розряди. Перший стовпчик вказує, що при формуванні другого перевірочного розряду беруть участь другій і третій інформаційні розряди. Позначають породжуючу матрицю, що написана в канонічній формі, наступним чином:

$$G(n,m) = \left\| R_{k \times m} \cdot I_m \right\| \quad \text{або} \quad G(n,m) = \left\| I_m \cdot R_{k \times m} \right\|.$$

Процес кодування математично записується в вигляді добутку матриці – рядка, що відображає кодову комбінацію, яка передається, на породжуючу матрицю. Припустимо, що кодова комбінація, яка передається, записана в вигляді вектору:

$$I = \left\| I_m I_{m-1} \ldots I_2 I_1 \right\|.$$

Тоді процес кодування запишеться в вигляді:

$$F = IG(n,m) = \left\| \underbrace{\Pi_k \Pi_{k-1} \ldots \Pi_1}_{\begin{matrix}\text{перевір.} \\ \text{розряди}\end{matrix}} \; \underbrace{I_m I_{m-1} I_{m-2} \ldots I_2 I_1}_{\begin{matrix}\text{інформ.} \\ \text{розряди}\end{matrix}} \right\|.$$

В результаті кодування отримаємо комбінацію, де перші m розрядів – інформаційні, а останні k розрядів – перевірочні Π_j.

Процес декодування математично описують добутком перевірочної матриці $H(n,m)$ і вектору-стовпчика, що відображає прийняту кодову комбінацію $F = F^T + E^T$, де E^T – вектор помилки

$$C = H(n,m)(F^T + E^T), \tag{4.1}$$

де:

C – результат декодування (синдром);

$(\ )^T$ – знак транспонування.

Оскільки зв'язок між породжуючою і перевірочною матрицями визначається рівністю:

$$G(n,m)\,H^T(n,m) = H(n,m)\,G^T(n,m) = 0,$$

то рівність (4.1) буде мати вигляд:

$$C = H(n,m)\,E^T = E\,H^T(n,m).$$

Якщо позначити h_i – i – й стовпчик перевірочної матриці, то: $H(n,m) = \|h_1 h_2 \ldots h_n\|$.

Вектор помилки ($E = \|0010\ldots010\ldots0\|$) містить **1** на тих позиціях, символи на яких викривлені. Нехай ці позиції мають номери $n_1, n_2, \ldots n_\ell$. Тоді буде справедливо рівність:

$$C = H(n,m)\,E^T = \sum_{i=1}^{\ell} h_{\nu_i}.$$

Отже, якщо необхідно виявити ℓ помилок, то повинно бути виконана умова:

$$\sum_{i=1}^{\ell} h_{\nu_i} \neq 0$$

при будь-якому поєднанні ℓ викривлених символів. Якщо необхідно виправити ℓ помилок, то сума: $\sum_{i=1}^{\ell} h_{\nu_i}$ для будь-яких ℓ конкретних стовпчиків перевірочної матриці повинна бути цілком певною, і не співпадати з аналогічною сумою для інших ℓ стовпчиків. Так, наприклад, при виправленні одинокої помилки всі стовпчики перевірочної матриці повинні бути різноманітні, т. т. $h_i \neq h_j$ при будь-яких i i j, $i \neq j$. Якщо необхідно виправити одиноку помилку і виявити пакет, який складається з трьох і двох символів, то необхідно виконати наступні умови:

– $h_i \neq h_j$ при $i \neq j$ (для виправлення одинокої помилки);

– $h_i + h_{i+1} \neq 0; h_i + h_{i+1} \neq h_j$ при будь-яких i i j (для виявлення пакету з двох помилок);

– $h_i + h_{i+1} + h_{i+2} \neq 0; h_i + h_{i+1} + h_{i+2} \neq h_j$ при будь-яких i i j (для виявлення пакету з трьох помилок).

Матриці $H(n,m)$ і $G(n,m)$ можна змінити ролями. Тоді матриця $H(n,m)$ буде породжуючою, а $G(n,m)$ – перевірочною. Коди, взаємозв'язані між собою таким чином, називають **дуальними** (подвійними).

8.3. Блочні лінійні коди. Коректуючи властивості блочних кодів. Коди з перевіркою на парність. Коди Хеммінга

Блочні лінійні коди

Коректуючи властивості блочних кодів

У цілому в процесі передачі можна розрізняти три ситуації: перекручування (помилки) відсутні, перекручування виявляються (прийнята заборонена комбінація) і перекручування не виявляються – одна дозволена комбінація переходить в іншу дозволену.

Говорять, що в каналі відбулася помилка кратності t, якщо в кодовій комбінації прийнято помилково t символів.

У результаті можна дійти висновку, що максимальне число помилок, що вірогідно виявляється в комбінації коригувального коду, пов'язане з кодовою відстанню наступним співвідношенням:

$$t_{виявл} = d_{min} - 1$$

При $d_{min} \le t_{виявл}+1$ або при $t \ge t_{виявл}$ прийнята кодова комбінація може виявитися дозволеною й помилка не буде виявлена. Однак і при $t \ge t_{виявл}$ можливі ситуації, коли комбінація виявляється забороненою, тобто частково будуть виявлятися й помилки кратності більшої, ніж $t_{виявл}$ (але це вже не вірогідно).

Як приклад розглянемо властивості, що виявляють, трьохразрядного коду. При $d_{min}=1$ дозволені комбінації – це всі комбінації коду 000, 001, 010, 011, 100, 101, 110, 111.

Будь-яка одиночна помилка трансформує кожну із цих комбінацій в іншу дозволену, отже достовірне виявлення помилок неможливо.

При $d_{min}=2$ підмножина дозволених комбінацій може бути утворена за принципом парності в них числа одиниць: 000, 011, 101, 110. До заборонених віднесемо інші комбінації: 001, 010, 100, 111. Жодна із заборонених кодових комбінацій при одиночній помилці не переходить в іншу дозволену комбінацію, тобто при $d_{min}=2$ код вірогідно виявляє всі одиночні помилки. Що стосується помилок більш високої кратності, то частина з них виявляється, а частина ні. Так виявляються всі трикратні помилки, однак дворазові помилки переводять одну дозволену кодову комбінацію в іншу й не можуть бути виявлені. Збільшення кодової відстані до $d_{min}=3$ дозволяють використовувати для передачі тільки дві кодові комбінації 000 і 111 або 001 і 110 і т.д., але зате тут уже й дворазова помилка не може перевести одну дозволену кодову комбінацію в іншу.

Модель трикратного коду можна використовувати й для пояснення принципів використання надмірності для виправлення помилок.

Основою методики, що викладається, є припущення про те, що чим більше кратність помилки, тим вона менше ймовірна. Це припущення гарно погоджується з біноміальною моделлю помилок у дискретному каналі зв'язку.

У цьому випадку при помилках найбільшу ймовірність будуть мати помилкові комбінації віддалені від переданої дозволеної на відстані d=1, менш імовірними будуть переходи в комбінації віддалені від переданої на відстані d=2 і т.д.

У результаті в процесі прийому-передачі кодових комбінацій по каналу зв'язку з помилками кожна з дозволених (переданих) комбінацій буде трансформуватися в деяку множину прийнятих кодових комбінацій, локалізованих у методиці Хемминга навколо переданої дозволеної кодової комбінації (якщо її відображати точкою відповідного n-мірного простору). Збільшуючи відстань між центрами цих областей в n-мірному просторі (точками, що відповідають дозволеним кодовим комбінаціям), можна реалізувати процедуру виправлення помилок заданої кратності, якщо ухвалювати рішення щодо декодування тієї дозволеної комбінації, відстань якої від прийнятої (забороненої) кодової комбінації виявляється мінімальним (декодування по найменшій відстані). Цей метод декодування є оптимальним для симетричного каналу (для інших може бути й не оптимальним)

Легко тепер зрозуміти, що максимальне число помилок, що **імовірно виправляються**, у комбінації коригувального коду пов'язане з кодовою відстанню коду нерівністю (якщо d_{min} – непарне):

$$t_{випр} \le (d_{min}-1)/2,$$

і, якщо d_{min} – парне:

$$t_{випр} \le d_{min}/2 - 1.$$

Дійсно, для виправлення помилок кратності t необхідно включити в кожну із зазначених вище підмножин всі заборонені кодові комбінації, що відстоять від дозволеної на відстань d=t. Для того, щоб сусідні підмножини не перетиналися, відстань між найближчими дозволеними комбінаціями повинне бути не менше 2t+1.

Таким чином, код має властивість **виявляти й виправляти t-кратні помилки**, якщо виконується умова

$$d_{min} \ge 2t+1 \qquad (4.2)$$

Звернемося знову до трьохразрядного коду. За дозволені комбінації приймемо 000 і 111. Тоді дозволеної комбінації 000 відповідає підмножина заборонених комбінацій 001, 010, 100, що утворяться за рахунок виникнення однократної помилки. Подібним же чином дозволеної комбінації 111 відповідає підмножина заборонених комбінацій 110, 101, 011.

При однократному перекручуванні кожної із двох дозволених комбінацій прийнята комбінація не вийде за межі “своєї” підмножини й помилка буде виправлена. Що стосується подвійних помилок, то вони виявляються, однак при їхньому виправленні приймаються невірні рішення. Потрійні помилки не виявляються взагалі.

Завадостійке кодування можна застосовувати й у дискретних каналах зі стиранням. Якщо в прийнятій комбінації перекручених символів немає, а є тільки $t_{ст}$ стертих (непізнаних) символів, то при виконанні умови $t_{ст} \le d_{min}-1$ ці символи можуть бути відновлені при декодуванні по мінімуму відстані. Той самий код може декодуватися з виправленням помилок і стирань. **У загальному випадку код з відстанню d_{min} виправляє будь-які $t \le t_{ст}$ стирань і будь-які $t \le t_{випр}$ помилок, якщо $d_{min} \ge 2t_{випр} + t_{ст} + 1$.**

Крім сполучень помилок і стирань, що виявляються й виправляються, кратність яких визначається кодовою відстанню, коди з корекцією часто виправляють або виявляють і багато інших сполучень помилок великої кратності.

Матеріал, розглянутий вище, показує важливу роль кодової відстані d_{min} коду як основного показника його здатності, що виявляє й виправляє. Збільшуючи довжину коду n і зберігаючи число дозволених кодових комбінацій, можна одержати як завгодно велике значення d_{min}. Але зі збільшенням n зменшується швидкість передачі повідомлень. Якщо довжина коду n задана, то будь-яке значення $d_{min} \leq n$ можна одержати, зменшуючи число дозволених комбінацій, але при цьому зменшується й кількість повідомлень, які можна закодувати. Отже, задачу пошуку найкращого коду необхідно формулювати так: при заданих n і N_p знайти код, що має можливо велике значення d_{min}.

Характеристикою коригувального коду, що вказує ступінь подовження кодової комбінації, є **надмірність**. Відносна надмірність може бути виражена співвідношенням:

$$\eta_m = \frac{n-m}{n}$$

відносна швидкість коду:

$$R_n = \frac{m}{n} = 1 - \eta_m$$

Коди, що забезпечують задану коригувальну здатність при мінімальній надмірності, називаються оптимальними.

Загального методу відшукання оптимальних кодів на данний час не існує.

Кодова відстань d_{min} повністю характеризує стійкість коду щодо заданого рівня перешкод. Однак у реальних умовах завадостійкість коду зручно описувати статистичними показниками. До їхнього числа ставляться: імовірність правильного прийому $P_{пр}$, імовірність помилки $P_{пом}$. Ці ймовірності можна визначити, знаючи структуру коду й статистику перешкод.

$$P_{пр} = \sum_{i=0}^{t} C_n^i p_0^i (1-p_0)^{n-i}; P_{ош} = \sum_{i=t+1}^{n} C_n^i p_0^i (1-p_0)^{n-i} = 1 - P_{пр}$$

Часто на перший план виходить не оптимальність коду, а складність технічної реалізації.

Таким чином, завадостійке кодування – найбільш діючий засіб збільшення вірогідності зв'язку.

Кодова відстань d_{min} коду є основним показником його що виявляє й виправляє здатності, при цьому максимальне число помилок, що виявляються вірогідно, у комбінації коригувального коду дорівнює $t_{виявл} = d_{min} - 1$, а що виправляються – $t_{випр} \leq (d_{min}-1)/2$.

Коди з перевіркою на парність

При побудові таких кодів послідовність розрядів, що передається розбивається на групи. В найбільш простому випадку перевірка на парність здійснюється в кожній групі, в результаті чого число одиниць в групі доводиться

до парного. Перевірочна матриця коду з перевіркою на парність має вигляд: $H(n, n-1) = 111...1$.

Основним недоліком коду є невиявлення помилок парної кратності. Тому такі коди знаходять застосування в тих ланках АСУ, де найбільш ймовірні одинокі помилки, наприклад, в ланці АПД – ЕОМ. Помилки ж, що виникають в каналі зв'язку, мають тенденцію до згуртовування. Для усунення цього недоліку групи розрядів (кодові комбінації) записуються в вигляді матриці. Після цього здійснюється перевірка на парність стовпчиків отриманої матриці.

За наявності однієї групової помилки довжиною не більш числа елементів рядку матриці в кожну перевірку буде входити не більш одного викривленого розряду (відбудеться декореляція помилок). Помилки не будуть виявлені, якщо викривлене число розрядів в стовпчику є парним.

Якщо помилки незалежні, то даний код еквівалентний коду з перевіркою на парність по рядкам. Якщо помилки корельовані, то за рахунок перевірки рознесених розрядів (за рахунок декореляції помилок) даний код буде більш завадостійким.

Для підвищення виявляючої спроможності, перевірка на парність може бути проведена водночас по стовпчикам і діагоналям або по рядкам і стовпчикам. Останній код називають **матричним**. В даному коді перевірочні розряди формуються по наступним правилам:

$$
\begin{array}{llll}
a_{11} \quad a_{12} \ldots a_{1q} & b_1 & b_i = a_{i1} + a_{i2} + \ldots + a_{iq}; \\
a_{21} \quad a_{22} \ldots a_{2q} & b_2 & c_j = a_{1j} + a_{2j} + \ldots + a_{\ell j}; \\
\ldots\ldots\ldots\ldots & \vdots & \\
a_{\ell 1} \quad a_{\ell 2} \ldots a_{\ell q} & b_\ell & c_{q+1} = c_1 + c_2 + \ldots + c_q; \\
\multicolumn{2}{c}{\text{---------}} & \text{або} \\
c_1 \quad c_2 \ldots c_q & c_{q+1} & c_{q+1} = b_1 + b_2 + \ldots + b_\ell .
\end{array}
$$

При такій побудові коду будуть виявлені всі однократні, двократні і трикратні помилки, а також всі непарні помилки і деякі парні помилки більшої кратності. Код знаходить застосування для виявлення помилок в накопичувачах на магнітних стрічках в ЕОМ.

Матричні коди можуть використовуватися в поєднанні з іншими кодами. При цьому кожний рядок матриці є дозволеною комбінацією будь якого коду. В цьому випадку говорять про добуток двох кодів. Добуток кодів використовується і для виправлення помилок.

Коди Хеммінга

Як показано вище, для виправлення одинокої помилки необхідно, щоб всі стовпчики перевірочної матриці були різноманітні. Якщо число перевірочних символів рівно k, то кількість різноманітних ненульових векторів (для двійкового коду) рівно $2^k - 1$. Коди, у яких довжина кодової комбінації визначається по вираженню $n = 2^k - 1$, число перевірочних розрядів рівно

$k = 3, 4 \ldots$, називають **кодами Хеммінга.** Часто до кодів **Хеммінга** відносять коди, у яких кожний стовпчик перевірочної матриці в двійковому коді відображає номер цього стовпчика.

Так, наприклад, код **Хеммінга (7,4)** має наступну перевірочну матрицю:

$$H(7,4) = \begin{matrix} 1\,2\,3\,4\,5\,6\,7 \\ \left\| \begin{matrix} 1\,0\,1\,0\,1\,0\,1 \\ 0\,1\,1\,0\,0\,1\,1 \\ 0\,0\,0\,1\,1\,1\,1 \end{matrix} \right\| \end{matrix}$$

При такій перевірочній матриці в випадку викривлення одного розряду в результаті декодування отримаємо номер позиції, на якій відбулася помилка. Перевірочні символи в коді **Хеммінга** встановлюються на позиції, що входять тільки в одну перевірку. Такими позиціями є 1, 2, 4.

Для виправлення одинокої і виявлення дворазової помилки додається перевірка на парність всіх позицій кодової комбінації. Перевірочна матриця в цьому випадку має вигляд:

$$H(7,4) = \left\| \begin{matrix} 1\,0\,1\,0\,1\,0\,1 \\ 0\,1\,1\,0\,0\,1\,1 \\ 0\,0\,0\,1\,1\,1\,1 \\ 1\,1\,1\,1\,1\,1\,1 \end{matrix} \right\|$$

Останній рядок цієї матриці забезпечує загальну перевірку на парність. Якщо відбудеться одинока помилка, то загальна перевірка на парність дасть **1**. Інші ж перевірки покажуть номер викривленого розряду. Якщо викривлені два розряди, то загальна перевірка на парність дасть **0**. Інші ж перевірки вкажуть на якусь позицію, але не на ту, на якій відбулися помилки.

Практична реалізація коду Хемінга

Теоретичні відомості. Розглянемо задачу побудови коду, що складається із 16-ти кодових слів виду $a_1a_2a_3a_4$, де a_i=0 або 1, в якому можливе виявлення та виправлення помилок.

Для розв'язку використаємо три допоміжні (перевірочні) символи, тобто кожне з 16-ти кодових слів кодуємо двійковим словом довжини 7: $a_1a_2a_3a_4a_5a_6a_7$. Нашу задачу можливо переформулювати, як визначення одного із чисел 0,1...7, які б вказували на місце похибки (0 – немає похибки). Накладемо умови (у вигляді рівностей за **mod 2**) на перевірочні символи:

$$a_5=a_2+a_3+a_4$$
$$a_6=a_1+a_3+a_4$$
$$a_7=a_1+a_2+a_4$$

Для виявлення похибки достатньо обчислити суму:

$$S_1=a_4+a_5+a_6+a_7.$$

Якщо $S_1=1$ то похибка є, інакше "немає".

При наявності похибки перевіримо, чи не міститься вона серед a_1 чи a_7. Для цього обчислимо $S_2=a_2+a_3+a_6+a_7$.

Якщо $S_1=S_2=1$ то похибка в a_6 або a_7.
Якщо $S_1=1$, $S_2=0$ то похибка в a_4 або a_5.
Якщо $S_1=0$, $S_2=1$ то похибка в a_2 або a_3.
Якщо $S_1=S_2=0$ то похибка в a_1 або немає.

Для визначення місця похибки слід обчислити суму

$$\mathbf{S_3=a_1+a_3+a_5+a_7.}$$

Тобто матимемо три перевірочні відношення

$$\mathbf{S_1=a_4+a_5+a_6+a_7 \neq 0}$$
$$\mathbf{S_2=a_3+a_2+a_6+a_7 \neq 0}$$
$$\mathbf{S_3=a_1+a_3+a_5+a_7 \neq 0}$$

які шляхом порівняння з 0(mod 2) "обчислюють" або відсутність похибки, або вказують її місце. Положення одиничної похибки визначається числом $\mathbf{S_1S_2S_3}$ в двійковій системі лічби. Таким чином маємо код (4,7) Хемінга довжина слова 7 та 4 інформаційні символи.

Якщо виявляти подвійну похибку, то слід виконати перевірку:

$$\mathbf{S_0=a_0+a_1+a_2+a_3+a_4+a_5+a_6+a_7 \neq 0}$$

де $\mathbf{a_0}$ – додатковий (четвертий) перевірочний символ, значення обчислємо за формулою:

$$\mathbf{a_0=a_1+a_2+a_3+a_4+a_5+a_6+a_7 \ (mod\ 2)}$$

Попередні $\mathbf{a_5, a_6, a_7}$ допомагали виявити та виправити одиночну похибку.

Якщо виникла подвійна похибка, то $\mathbf{S_0=0(mod\ 2)}$ та хоча б одне із чисел $\mathbf{S_1,S_2,S_3}$ буде не нульовим. Виправлення такої похибки неможливе. Побудована множина кодових слів зветься розширеним кодом (4,8) Хемінга (довжина слова 8 та з них інформаційними символами).

8.4. Циклічні коди. Засоби опису циклічних кодів. Властивості циклічних кодів по виявленню помилок. Укорочені циклічні коди. Коди Боуза-Чоудхурі-Хоквінгема. Коди Ріда-Соломона. Код Файра

Засоби опису циклічних кодів

Розглянемо циклічні коди. До них відносяться:

– Укорочені циклічні коди.

– Коди Боуза-Чоудхурі-Хоквінгема.

– Коди Ріда-Соломона.

– Код Файра.

Назва цього класу кодів відбулася від основної з властивостей, яка полягає в тому, що результатом порозрядної перестановки (циклічного зсуву) дозволеної кодової комбінації стає також дозволена комбінація.

Циклічні коди зручно описувати багаточленами змінної x. При цьому показники ступеня відповідають номерам розрядів, а коефіцієнтами цього багаточлену є **0** і **1** кодової комбінації, що відображається. Наприклад, комбінацію 1001101 можна записати в вигляді: $x^6 + x^3 + x^2 + 1$.

В теорії циклічних кодів операції над кодовими комбінаціями зводяться до алгебраїчних операцій з отриманими багаточленами в відповідності з законами звичайної алгебри багаточленів, за винятком того, що складання і віднімання замінюється складанням за модулем **2**.

Принцип виявлення помилок циклічним кодом полягає в наступному. В якості дозволених приймаються такі комбінації, що без залишку діляться на заздалегідь вибрану комбінацію, яка відображається породжуючим (утворюючим) багаточленом $P(x)$ ступеня k. Прийнята кодова комбінація підлягає діленню на $P(x)$.

Якщо комбінація викривлена, то утвориться залишок, не рівний нулю, що і забезпечує виявлення помилок.

При кодуванні багаточлен $G(x)$, відповідний m – розрядній інформаційній комбінації беззайвого коду, умножається на x^k, що підвищує ступінь багаточлену від $m-1$ до $n-1$. Після цього добуток $G(x)\,x^k$ ділиться на породжуючий багаточлен $P(x)$ і залишок від ділення $R(x)$ складеться з добутком $G(x)\,x^k$. В результаті одержується кодовий багаточлен $F(x) = G(x)\,x^k + R(x)$, відповідний комбінації циклічного коду.

Операція складання $G(x)\,x^k + R(x)$ означає приписування перевірочних символів на відведені для них k позицій.

Отриманий багаточлен $F(x)$ відповідає дозволеній кодовій комбінації, так як він ділиться без залишку на утворюючий багаточлен $P(x)$. Справді, оскільки $G(x)\,x^k = \varphi(x)\,P(x) + R(x)$, де $\varphi(x)$- частка від ділення $G(x)\,x^k$ на $P(x)$, то справедлива (з урахуванням заміни алгебраїчного складання на складання за модулем **2**) рівність:

$$G(x)\,x^k + R(x) = \varphi(x)\,P(x) = F(x),$$

що і вимагалося довести.

Прийняту комбінацію, яку позначимо $F'(x)$, можна уявити в вигляді двох доданків: переданого кодового багаточлену $F(x)$ і багаточлену помилки $E(x)$, т. т. $F'(x) = F(x) + E(x)$. Цей багаточлен підлягає діленню на $P(x)$, і по наявності залишку приймається рішення про вірність прийнятої комбінації. Якщо ділення здійснюється без залишку, то приймається рішення про то, що інформація не викривлена.

Циклічні коди можуть задаватися перевірочними або породжуючими матрицями. Так, кожний стовпчик канонічної форми перевірочної матриці можна визначити шляхом знаходження залишків від ділення одночлену $x^i (0 \le i \le n-1)$ на багаточлен $P(x)$.

Наприклад, перевірочна матриця $H(7,4)$ буде мати вигляд:

$$\begin{matrix} & 1 & 2 & 3 & 4 & 5 & 6 & 7 \end{matrix}$$

$$H(7,4) = \|h_1 h_2 h_3 h_4 h_5 h_6 h_7\| = \begin{Vmatrix} 1 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{Vmatrix} \qquad (4.3)$$

В деяких джерелах інформації перевірочна матриця циклічного коду визначається через так звану **матрицю переходів** S. Ця матриця визначається видом утворюючого багаточлену і засобом побудови кодуючих і декодуючих пристроїв. Так, ця матриця може мати наступний вигляд:

$$S = \begin{Vmatrix} 0 & 0 & 0 & 0 & \dots & 0 & a_1 \\ 1 & 0 & 0 & 0 & \dots & 0 & a_2 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & 0 & \dots & 1 & a_k \end{Vmatrix},$$

де a_k – коефіцієнти в вираженні утворюючого багаточлену.

Наприклад: Написати матрицю переходів при $P(x) = x^3 + x + 1$. Багаточлен $P(x)$ можна написати в вигляді:

$$P(x) = a_4 x^3 + a_3 x^2 + a_2 x + a_1 x^0$$

при $a_4 = 1; a_3 = 0; a_2 = 1; a_1 = 1$. Тоді матриця переходів буде мати вигляд:

$$S = \begin{Vmatrix} 0 & 0 & 1 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{Vmatrix} \qquad (4.4)$$

Кожний стовпчик перевірочної матриці $h_j^T = \|h_{1j}h_{2j}\ldots h_{kj}\|$ одержується шляхом перемноження матриці S^{j-1} на перший стовпчик матриці H, т. т. $h_j = S^{j-1}h_1$.

Зазвичай в якості першого стовпчика береться матриця $h_1^T = \|100\ldots0\|$. Тоді перевірочну матрицю можна записати в вигляді:

$$H(n,m) = \|I_1, Sh_1, S^2h_1, \ldots, S^{n-1}h_1\| = \|I, S, S^2, \ldots, S^{n-1}\|h_1,$$

де I – одинична матриця.

В результаті декодування отримаємо:

$$C = H(n,m)[п_1п_2п_3\ldots п_kC_mC_{m-1}\ldots C_1]^T = \\ = \|I\,п_1 + Sп_2 + S^2п_3 + \ldots + S^{k-1}п_k + S^kC_m + \ldots + S^{n-1}C_1\|h_1. \qquad (4.5)$$

Властивості циклічних кодів по виявленню помилок

1. Якщо утворюючий багаточлен містить більш одного члена, то циклічний код виявляє всі одиночні помилки. При поданні циклічного коду багаточленами одиночна помилка, описується одночленом $E(x) = x^i$, де i вказує номер викривленого розряду $(0 \le i \le n-1)$. Оскільки одночлен не ділиться на багаточлен без залишку, то помилка буде виявлена.

2. Циклічний код з утворюючим багаточленом $P(x) = x+1$ виявляє всі непарні помилки. Використовуючи правила побудови перевірочної матриці для $P(x) = x+1$, отримаємо: $H = \|111\ldots1\|$.

При такій перевірочній матриці залишок визначається сумою за модулем **2** всіх елементів прийнятої кодової комбінації (перевірка на парність). Тому все викривлення на непарній кількості позицій будуть виявлені. Доказати цю властивість можна і іншим засобом. Відомо, що помилки непарної кратності відображаються багаточленом:

$$E(x) = x^{\alpha}(x^i + \ldots + x^j + 1),$$

в якому число ненульових членів не парно. Так як при діленні такого багаточлена $E(x)$ на багаточлен $P(x) = x+1$ залишок завжди буде рівний **1**, то означені помилки будуть виявлені.

3. Циклічний код виявляє всі одиночні і дворазові помилки, якщо розрядність коду n не більше довжини циклу $\ell_ц$ утворюючого багаточлена, який використовується, т. т. $n \le \ell_ц$. Під **довжиною циклу багаточлена** розуміють мінімальний показник ступеня двочлена $x^{\ell_ц}+1$, при якому цей двочлен ділиться без залишку на утворюючий багаточлен $P(x)$. Дворазова помилка описується двочленом вигляду:

$$E(x) = x^i + x^j = x^j(x^{i-j}+1),$$

де $i \ne j$, $1 \le i-j \le n-1$.

При $n \le \ell_ц$ завжди справедлива нерівність $i - j < \ell_ц$, бо $(i-j)_{макс} = n-1$. З визначення довжини циклу слідує, що двочлен $E(x)$ не ділиться без залишку на $P(x)$. Якщо довжина кодової комбінації більше довжини циклу, т. т. $n > \ell_ц$, то все викривлення, котрі відображаються багаточленом помилки $E(x) = C(x)(x^{\ell_ц + 1})$, не будуть виявлені. Тут $C(x)$ - багаточлен будь-якого ступеня. Отже, при вибраному багаточлену P(x) ступеня k з довжиною циклу $\ell_ц$ число інформаційних символів в кодовій комбінації повинно задовольняти співвідношенню $m \le \ell_ц - k$ (так як $n = m + k$).

4. Циклічний код з багаточленом $P(x)$ ступеня k виявляє всі групові помилки тривалістю в k розрядів і менш. Будь-яка помилка в k розрядів описується багаточленом ступеня $k-1$, т. т.:

$$E(x) = x^i(x^{k-1} + x^{k-2} + \ldots + 1).$$

Багаточлен ступеня $k-1$ на багаточлени ступеню k не ділиться і, таким чином, помилка виявляється.

5. Циклічний код з утворюючим багаточленом $P(x)$ ступеня k не виявляє $\frac{1}{2^{k-1}}$ частину помилок $k+1$- й кратності. В розглядуваному випадку багаточлен помилки має вигляд:

$$E(x) = x^i\, E_1(x)$$

де $(0 \le i \le m)$, причому $E_1(x)$ має ступінь k. Загальне число можливих багаточленів ступеня k рівно 2^{k-1}. З цих багаточленів лише один співпадає по вигляду з $P(x)$ і ділиться на нього без залишку, а інші – не діляться. Отже, $\frac{1}{2^{k-1}}$ частина розглядуваних помилок не виявляється.

Приклад: Використовується циклічний код з утворюючим багаточленом: $P(x) = x^3 + x^2 + 1$.

Помилки четвертої $(k+1=4)$ кратності описуються чотирма багаточленами $(2^{k-1} = 2^2 = 4)$: $x^3 + x^2 + x + 1;\ x^3 + x^2 + 1;\ x^3 + x + 1;\ x^3 + 1$.

З усіх чотирьох варіантів помилок не виявляється помилка, що описується багаточленом: $E_1(x) = x^3 + x^2 + 1$.

Отже, відносна частина помилок, що не виявляються, четвертої кратності рівна: $\frac{1}{2^{k-1}} = \frac{1}{2^{3-1}} = \frac{1}{4}$.

6. Циклічний код з утворюючим багаточленом $P(x)$ ступеня k не виявляє $\frac{1}{2^k}$ частину помилок більш $k+1$- й кратності. Помилки кратності більш $k+1$ описуються багаточленом: $E(x) = x^i E_1(x)$,

де $E(x)$ – багаточлен ступеня $k+s;\ i = 1, 2, 3, \ldots;\ s = 1, 2, 3, \ldots$.

При діленні будь-якого з багаточленів ступеня k+s утворяться різноманітного вигляду залишки, ступінь яких не перевищує $k-1$. З загального числа залишків, рівного 2^k, тільки один дорівнює нулю, що свідчить про невиявлення помилки. Отже, відносна частина помилок, що не виявляються, кратності вище $k+1$ рівна $\frac{1}{2^k}$.

Аналізуючи перераховані властивості циклічного коду, можна побачити, що хист коду по виявленню і виправленню помилок повністю визначається вибраним утворюючим багаточленом $P(x)$. При виявленні помилок стандартні багаточлени мають вигляд:

$$P(x) = x^8 + x^2 + x + 1 \Rightarrow (d = 4) \text{ при довжині кодової комбінації } n \le 2^7;$$

$$P(x) = x^{16} + x^{15} + x^{13} + x^{11} + x^5 + x^3 + x + 1 \Rightarrow (d = 6) \text{ при } n \le 2^7;$$

$$P(x) = x^{16} + x^{12} + x^5 + 1 \Rightarrow (d = 4) \text{ при } n \le 2^{15}.$$

Розроблений ряд методик по вибору породжуючого багаточлену $P(x)$. В літературі коди називають по прізвищам вчених, що запропонували ту або іншу методику. Так отримали свою назву коди **Боуза-Чоудхурі-Хоквінгема** (БЧХ), коди **Ріда-Соломона**, коди **Файра** та інші.

Укорочені циклічні коди

Циклічні коди припускають рівність розрядності кодової комбінації, що формується, довжині циклу утворюючого багаточлена, т. т. $n = \ell_ц$. Однак ця вимога на практиці важко здійснюється, оскільки розрядність комбінацій, що видаються джерелом інформації, нерідко відрізняється від доцільної, в результаті чого $n < \ell_ц$. В цих випадках циклічний код перетворюють і використають так званий **скорочений циклічний код**, що одержується з циклічного шляхом винятку в ньому певного числа розрядів.

При цьому число стовпчиків перевірочної матриці $H(n,m)$ зменшується на величину, яка дорівнює числу інформаційних розрядів, що виключаються, причому виключаються стовпчики з найвищими номерами. Властивості коду по виявленню помилок при цьому не змінюються.

Коди Боуза-Чоудхурі-Хоквінгема

Коди Боуза-Чоудхурі-Хоквінгема (БЧХ) відносяться до класу циклічних і призначені для виправлення t_u-кратних помилок в n-розрядної кодової комбінації. Породжуючий багаточлен $P(x)$ визначається вираженням:

$$P(x) = \text{нзк}\{f_1(x), f_2(x), \ldots, f_{2t_u}(x)\},$$

де нзк – найменше загальне кратне; $f_i(x)$ – мінімальні багаточлени коренів $P(x)$ При цьому $f_1(x)$ мінімальний багаточлен кореня α з довжиною циклу $\ell_ц = n$;

$f_2(x)$ – мінімальний багаточлен кореня α^2; $f_i(x)$ – мінімальний багаточлен кореня α^i(гляди додаток 1).

Оскільки α^i (при парному i) є коренем того же багаточлена, коренем якого є α, то вираження для породжуючого багаточлена буде мати вигляд:

$$P(x) = \prod_{i=1,3,5}^{2t_u-1} f_i(x).$$

Багаточлен $f_1(x)$ визначається за формулою:

$$f_1(x) = (x - \alpha) \prod_{j=1,2,4}^{\nu} (x - \alpha^{2j}),$$

де ν визначається з умови $2(\nu+2) - \ell_ц = 1.$

Наприклад, для $f_1(x) = x^3 + x + 1$ елемент $\alpha = 010$ є його коренем. Елементи α^2, α^4 також будуть його коренями. Елемент α^8 буде рівний α^1, так як довжина циклу багаточлена $f_1(x)$ рівна 7. Отже,

$$f_1(x) = (x - \alpha)(x - \alpha^2)(x - \alpha^4).$$

Розкривши дужки, отримаємо:

$$f_1(x) = x^3 + x + 1.$$

Багаточлен $f_3(x)$ визначається по вираженню:

$$f_3(x) = (x - \alpha^3) \prod_{j=1,2,4}^{\nu} (x - \alpha^{3\cdot 2j}),$$

де ν визначається з умови $2(\nu+2) - j\ell_ц = 3;\ j = 1,2,3$ і т. д.

Приклад: Визначити породжуючий багаточлен для коду БЧХ, що виправляє помилки кратності до двох включно. Довжина кодової комбінації $n = 15$.

Виберемо неприводимий над полем $GF(2)$ багаточлен $f_1(x) = x^4 + x + 1$, довжина циклу якого рівна $\ell_ц = 15$. Його корені є елементи поля $GF(2^4)$. Примітивний корінь – α. Елементи α^2, α^4 також є корні багаточлену $f_1(x)$. Так як повинні бути виправлені дворазові помилки, то породжуючий багаточлен рівний:

$$P(x) = f_1(x)\, f_3(x).$$

Елемент α^3 є коренем багаточлена $f_3(x)$. Коренями цього багаточлена є також елементи $\alpha^6, \alpha^{12}, \alpha^{24} = \alpha^9$.

Отже:

$$f_2(x) = (x - \alpha^3)(x - \alpha^6)(x - \alpha^9)(x - \alpha^{12}).$$

Враховуючи, що $\alpha^4 + \alpha + 1 = 0$ і $\alpha^{15} = 1$ розкривши дужки, отримаємо:

$$f_2(x) = x^4 + x^3 + x^2 + x + 1.$$

Тоді породжуючий багаточлен буде рівний:

$$P(x) = f_1(x)\, f_2(x) = x^8 + x^7 + x^5 + x^4 + 1.$$

В результаті коренями багаточлена $P(x)$ є:

$$\alpha, \alpha^2, \alpha^4, (\alpha)^3, (\alpha^2)^4, (\alpha^4)^3, (\alpha^5)^3.$$

Якщо необхідно виправити не двократні помилки, а помилки кратності до t_u включно, то корні необхідно зводити в ступінь до $2t_u - 1$.

Для загального випадку існує така теорема. Якщо α – примітивний елемент поля $GF(2^k)$, то код, породжений багаточленом $P(x)$, є БЧХ – кодом тоді і тільки тоді, коли елементи: $\alpha, \alpha^3, \alpha^5, \ldots, \alpha^{2t_u - 1}$ є коренями цього багаточлена.

Ґрунтуючись на цій теоремі, можна запропонувати наступну методику вибору багаточлена $P(x)$ для коду БЧХ.

Задана довжина кодової комбінації $n = 2^z - 1$.

1. Двочлен $2^{2^{Z-1}} + 1$ завжди розкладається на неприводимі багаточлени у полі $GF(2)$.

2. З отриманих співмножників вибирають багаточлен $f_1(x)$ ступеня Z і довжиною циклу $\ell_{ö} = n$. Шляхом ділення $x^i (i = 1, \ldots, n-1)$ на цей багаточлен знаходять всі корені двочлену $x^{2^z - 1} + 1$.

3. Визначають приналежність коренів до отриманих при розкладі на множники багаточленам.

4. Корні багаточлену $f_1(x)$ зводять в непарні ступені від **1** до $2t_u - 1$, в результаті одержують набір корнів, що повинен мати утворюючий багаточлен $P(x)$.

5. Виявляють багаточлени, яким належать отримані корні.

6. Вибрані багаточлени перемножують і одержують вираз для $P(x)$.

Приклад: Визначимо $P(x)$ для коду БЧХ при $n = 15 = 2^4 - 1$ і $t_u = 3$.

1. Розкладемо на множники двочлен:

$$x^{2^4 - 1} + 1 = f_1(x)\, f_2(x)\, f_3(x)\, f_4(x)\, f_5(x) =$$

$$= (x+1)(x^2 + x + 1)(x^4 + x + 1)(x^4 + x^3 + 1) \cdot (x^4 + x^3 + x^2 + x + 1).$$

2. З співмножників вибираємо неприводимий багаточлен $f_3(x) = x^4 + x + 1$ і шляхом ділення $x^i (i = 1, 2, \ldots, 14)$ на $f_3(x)$ визначаємо всі корені двочлену $x^{2^4 - 1} + 1$.

3. Визначаємо приналежність коренів до співмножників.

Корні $\alpha^1, \alpha^2, \alpha^4, \alpha^8$ належать до $f_3(x)$.

4. Зводимо корені багаточлена $f_3(x)$ в ступені від **1** до $2 \cdot 3 - 1 = 5$.

Так як $\alpha^1, \alpha^2, \alpha^4, \alpha^8$ є коренями багаточлена: $f_3(x) = x^4 + x + 1$; $\alpha^3, \alpha^6, \alpha^9, \alpha^{12}$ є коренями багаточлена: $f_5(x) = x^4 + x^3 + x^2 + x + 1$; α^5, α^{10} є коренями багаточлена: $f_2(x) = x^2 + x + 1$, то:

$$P(x) = f_2(x)\, f_3(x)\, f_5(x) = x^{10} + x^8 + x^5 + x^4 + x^2 + x + 1.$$

Коди Ріда-Соломона

Коди **Ріда-Соломона** є недвійковим підкласом кодів БЧХ. Вони визначені над полем $GF(q) = GF(2^k)$. Довжина кодової комбінації рівна: $n = q - 1$. Породжуючий багаточлен визначається за формулою:

$$P(x) = (x - \alpha)(x - \alpha^2)\ldots(x - \alpha^{2t_u}),$$

де α – примітивний елемент поля $GF(q)$.

В результаті одержується код з $d - 1$ перевірочними символами.

Приклад: Виберемо $P(x)$ для коду **Ріда-Соломона** з $n = 7$, що виправляє дворазові і однократні помилки. Нехай поле $GF(8)$ задане табл. П.1. Тоді:

$$P(x) = (x - \alpha)(x - \alpha^2)(x - \alpha^3)(x - \alpha^4) = x^4 + \alpha^3 x^3 + x^2 + \alpha x + \alpha^3.$$

Слідує відзначити, що коефіцієнти багаточлена є елементами з поля $GF(8)$.

Приклад: Виберемо $P(x)$ для коду **Ріда-Соломона** з $n = 15$ і $t_u = 2$. Поле $GF(16)$ задане табл. П.3. Тоді:

$$P(x) = (x - \alpha)(x - \alpha^2)(x - \alpha^3)(x - \alpha^4) = x^4 + \alpha^{13} x^3 + \alpha^6 x^2 + \alpha^3 x + \alpha^{10}.$$

Таким чином, в коді **Ріда-Соломона** коефіцієнти при змінних не є двійковими числами, як це було в коді БЧХ, що дозволяє здійснити кодування недвійкової інформації.

Код Файра

Код Файра – це циклічний код, що забезпечує виправлення пакетів помилок тривалістю b та менш розрядів. Породжуючий багаточлен даного коду визначається за формулою:

$$P(x) = (x^{2b-1} + 1)\, P_1(x),$$

де $P_1(x)$ – примітивний багаточлен над полем $GF(q)$, ступінь якого не менше довжини пакету помилок b, що виправляється, і який не ділить двочлен $x^{2b-1} + 1$ без залишку.

Довжина кодової комбінації приймається рівної $n = \ell_ц(2b - 1)$, де $\ell_ц$ – довжина циклу багаточлена $P_1(x)$. Наприклад, $P(x) = x^4 + x + 1$ багаточлен, що не приводиться, в полі $GF(2)$, довжина циклу цього багаточлену $\ell_ц = 15$. Нехай $b = 4$. Очевидно, що $2b - 1 = 7$ не ділиться на 13. Тоді утворюючий багаточлен буде мати вигляд:

$$P(x) = (x^4 + x + 1)(x^7 + 1) = x^{11} + x^8 + x^7 + x^4 + x + 1.$$

Перемноження $P_1(x)$ на двочлен $x^{2b-1} + 1$ являє собою введення додаткової перевірки кодової комбінації на парність, що дасть в деякому роді зображення пакету помилок. Який-небудь пакет помилок тривалістю b та менш символів буде залишати по меншій мірі $b-1$ перевірок не заторкнутими, і тому можна визначити, який символ стоїть в початку пакету. Положення пакету помилок в кодовій комбінації визначається за рахунок багаточлена $P_1(x)$.