Міністерство освіти і науки України
Луцький національний технічний університет

ТЕОРІЯ ІНФОРМАЦІЇ ТА КОДУВАННЯ

Конспект лекцій
для здобувачів першого (бакалаврського) рівня вищої освіти
освітньої програми «Інформаційні системи та технології охорони і безпеки»
спеціальності 126 (F6) Інформаційні системи та технології
галузі знань 12 (F) Інформаційні технології
денної та заочної форм навчання

Луцьк 2026

УДК 621.391(075.8)
Т 33

Рекомендовано до видання вченою радою факультету КІТ ЛНТУ,
протокол № _____ від «___ » _____ 20 26 року.

Голова вченої ради факультету КІТ _____ Інна КОНДІУС

Електронна копія друкованого видання передана для внесення в репозитарій ЛНТУ
Директор бібліотеки _____ Наталія ПОЛІЩУК

Розглянуто і схвалено на засіданні кафедри комп'ютерної інженерії та безпеки ЛНТУ, протокол № _____ від « _____ » _____ 20 26 року.

Завідувач кафедри КІБ _____ Тарас ТЕРЛЕЦЬКИЙ

Укладачі: _____ Сергій ГРИНЮК, кандидат технічних наук,
Доцент кафедри комп'ютерної інженерії та безпеки ЛНТУ

_____ Микола ПОЛІЩУК, кандидат технічних наук,
доцент кафедри комп'ютерної інженерії та безпеки ЛНТУ

Рецензент: _____ Андрій ЯЩУК, кандидат технічних наук,
доцент кафедри комп'ютерної інженерії та безпеки ЛНТУ

Відповідальний за випуск: _____ Тарас ТЕРЛЕЦЬКИЙ, кандидат технічних наук, доцент кафедри комп'ютерної інженерії та безпеки ЛНТУ

Т 33 **Теорія інформації та кодування**: конспект лекцій для здобувачів першого (бакалаврського) рівня вищої освіти освітньої програми «Інформаційні системи та технології охорони і безпеки» галузі знань F (12) Інформаційні технології спеціальності F6 (126) Інформаційні системи та технології денної та заочної форм навчання / уклад. С.В. Гринюк, М.М. Поліщук. Луцьк: ЛНТУ, 2026. 88 с.

Конспект лекцій складений відповідно до робочої програми дисципліни «Теорія інформації та кодування» з метою використання студентами технічних спеціальностей при вивченні даного предмету.

ЗМІСТ

Тема 1. Інформація та інформаційні процеси

План

1. **Основні поняття теорії інформації та кодування**
2. **Моделі інформаційних систем**
3. **Види інформації. Теорема дискретизації**
4. **Предмет теорії інформації та кодування**

1. Базові поняття теорії інформації

У найзагальнішому розумінні інформація – це передавання, відображення певного різноманіття. В теорії інформації інформацію розглядають як кількісну міру усунення невизначеності, що зменшується у результаті отримання якихось відомостей.

Отже, ***інформація*** – це об'єктивно існуючий зміст, який характеризує стан і поведінку певної системи загалом або її окремих елементів та зменшує ступінь невизначеності у процесі його пізнання і переробки. Інформація протилежна невизначеності.

Виробничі процеси, а також процеси в природному середовищі пов’язані з передачею, одержанням, перетворенням, нагромадженням, зберіганням і відображенням інформації.

При цьому існують різні визначення інформації.

Як правило, під *інформацією* в широкому сенсі розуміють нові відомості про навколишній світ, які ми одержуємо в результаті взаємодії з ним, пристосування до нього й зміни його в процесі пристосування.

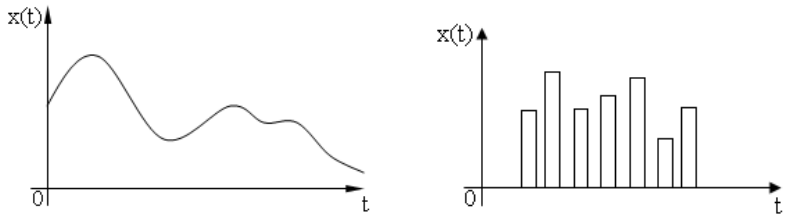
Найбільш узагальнюючим є таке визначення інформації. *Інформація* - це відомості, які є об’єктом зберігання, передачі й перетворення.

У найзагальнішому розумінні інформація – це передавання, відображення певного різноманіття. В теорії інформації інформацію розглядають як кількісну міру усунення невизначеності, що зменшується у результаті отримання якихось відомостей.

Потрібно розрізняти поняття «інформація» і «повідомлення».

Повідомлення - це форма подання інформації. Наприклад, при телеграфній передачі повідомленням є текст телеграми, що являє собою послідовність різних символів. При розмові пові-домлення являє собою механічні коливання з різною частотою й інтенсивністю голосових зв’язок людини. При телевізійних чорно-білих передачах повідомлення є зміною в часі яскравості елементів переданого зображення. При кольорових телевізійних передачах повідомлення також подається зміною кольору елементів зображення і т.д.

Усі повідомлення за характером змінюються в часі і їх поділяють на неперервні і дискретні. Неперервні в часі повідомлення відображаються неперервною функцією часу, рис.1.1.



а) б)

Рисунок 1.1

Дискретні повідомлення характеризуються тим, що вони надходять у певні моменти часу і описуються дискретною функцією часу, рис.1.1.

Кодування - перетворення інформації на впорядкований набір символів, елементів, знаків. При кодуванні кожному повідомленню з деякої множини, що називається ***ансамблем повідомлень***, ставиться у відповідність зумовлена ***кодова комбінація*** - набір символів (елементів, знаків). Множина повідомлень називається ***алфавітом повідомлень***, або ***первинним алфавітом***, а множина символів (елементів, знаків) називається ***алфавітом***

джерела, або ***вторинним алфавітом***. Побудована відповідно до певної схеми кодування множина кодових комбінацій називається ***кодом***. Залежно від алфавіту, що використовується для побудови кодових комбінацій, розрізняють ***двійкові (бінарні) коди***, алфавіт яких складається з двох символів: **0** і **1** і ***недвійкові (багатопозиційні, q-коди)***, алфавіт яких містить більшу кількість символів.

За функціональним призначенням коди поділяють на ***безнадмірні (некоригувальні***, ***первинні***, ***прості)*** і ***надмірні (коригувальні, завадостійкі)***. Перша група кодів призначена для ***економного кодування інформації – стиснення***. Друга використовується для виявлення та/чи виправлення помилок, що виникають у процесі передачі даних ***каналом зв'язку із завадами.***

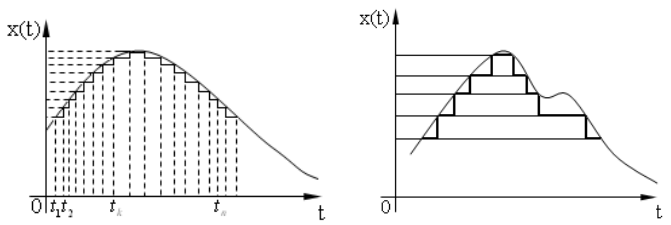
Для перетворення первинного сигналу до вигляду, придатного для використання в лінії зв'язку, використовується передавач (**модулятор**). У передавальному пристрої відбувається дія (вплив) на один або декілька параметрів носія за законом, який прийнятий при кодуванні повідомлень. Цей процес називають **модуляцією**, а модульовані параметри – інформативними.

Як правило, при передачі на сигнали впливають перешкоди. Під перешкодами розуміють вплив атмосферних перешкод або вплив сторонніх джерел сигналів, а також зміни сигналів у самій апаратурі (апаратурні перешкоди). Усі ці перешкоди викликають випадкові відхилення прийнятого повідомлення від переданого.

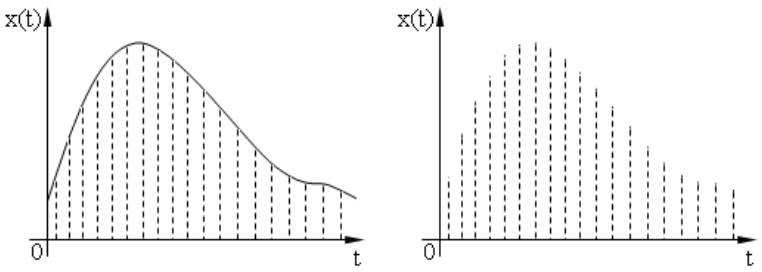
На боці прийому інформації відбувається відновлення за прийнятим сигналом переданого повідомлення. Для цього спочатку проводять **демодуляцію** сигналу, внаслідок чого відновлюється прийнятий сигнал переданого повідомлення. Під впливом перешкод прийнятий сигнал може значно відрізнятися від переданого. Тому для його відновлення у **розв'язувальному** пристрої відбувається обробка отриманого з лінії зв'язку сигналу з метою найбільш повного відтворення тієї інформації, яка передавалася на передавальному пристрої. Процедури обробки сигналів у розв'язувальному пристрої різні: фільтрація; обмеження; інтеграція; сума сигналів і т. п. Після обробки сигналів відбувається їх декодування, тобто перетворення сигналів на повідомлення, яке надходить одержувачу. На практиці часто необхідно забезпечити незалежну передачу повідомлень від декількох джерел. Але використання різних каналів зв'язку при цьому є економічно невигідним. Тому виникає завдання побудови систем, в якій використовують одну лінію зв'язку для передачі повідомлень від різних джерел. Такі системи називають багатоканальними. У багатоканальних системах на передавальному боці необхідно мати формувач канальних ознак (пристрій ущільнення), а на приймальному боці – пристрій розділення сигналів.

Сигнали за своєю структурою у каналі зв'язку повинні поділятися на: неперервні за рівнем і часом; дискретні за рівнем і часом; дискретні за рівнем і неперервні за часом; неперервні за рівнем і дискретні за часом.

Сигнали, неперервні за рівнем і часом, називають неперервними, вони мають вигляд, наведений на рис.1.1а. Сигнали, дискретні за рівнем і часом, називають дискретними і вони мають вигляд, наведений на рис. 1.2а. Квантування за рівнем і часом є заміною неперервного сигналу *X (t),* рис.1.21а, безліччю його дискретних значень у фіксовані моменти часу, які відрізняються між собою, рис. 1.3а.



а) б)



в) г)

Рисунок 1.2

Сигнали, дискретні за рівнем і неперервні за часом, називають дискретно-неперервними, рис.1.2б. Сигнали, неперервні за рівнем і дискретні за часом, називають дискретно-неперервними, рис.1.2в. На практиці з цієї групи сигналів найбільше застосування мають сигнали, неперервні за рівнем і дискретні за часом, оскільки вони найбільш просто реалізуються в технічних пристроях. Передача сигналів будь-яким реальним каналом зв'язку завжди пов'язана з граничною частотою пропускання, перевищення якої призводить до спотворення сигналів. А якщо необхідно передати каналом зв'язку неперервний сигнал, то його попередньо квантують із періодом квантування $T_k = \frac{1}{2f_c}$, який дає можливість потім точно відтворити початковий неперервний сигнал, рис. 1.3г.

Коди можуть бути ***рівномірними*** і ***нерівномірними*** - з постійною і змінною кількістю розрядів.

Канал зв'язку - це середовище передачі інформації, що характеризується максимально можливою для нього швидкістю передачі даних – ***пропускною здатністю***, або ***ємністю*** каналу.

Пропускну здатність каналу зв'язку без шуму можна наближено обчислити, знаючи максимальну частоту хвильових процесів, допустимих у цьому каналі. Вважається, що швидкість передачі даних може бути не менше цієї частоти. Типові канали зв'язку: телеграфний, телефонний, оптоволоконний, цифровий телефонний. Найбільш поширені телефонні лінії зв'язку, для яких досягнута швидкість передачі даних, >50 *Кбод*.

Сигнал – це матеріальний переносник повідомлення. Сигнали можуть бути: електричні; електромагнітні; світлові; механічні; звукові; ультразвукові і т.д. Для передачі повідомлень необхідно застосовувати той переносник, який може ефективно поширюватися при використанні лінії зв'язку. Наприклад, по електричній лінії зв'язку найбільш легко проходить постійний струм і змінні струми невисоких частот (не більше кількох десятків кГц) з використанням радіолінії ефективно поширюються тільки електромагнітні коливання високих частот (від 100 кГц до 10 тис. мГц).

Будь-який сигнал характеризується такими основними параметрами: тривалістю, шириною частотного спектра та динамічним діапазоном.

Під *тривалістю* T_c сигналу розуміють час, протягом якого він знаходиться в каналі зв'язку. *Частотний спектр* F_c сигналу визначає смугу частот, яку він охоплює під час передачі в каналі зв'язку. Залежно від виду сигналу (аналоговий, дискретний) частотний спектр може бути і нескінченним; тому на практиці його обмежують для можливості передачі в каналах з обмеженою смугою частот. Так, телефонні розмови ведуться в каналах зі смугою пропускання 3100 Гц (300...3400 Гц), хоча сам початковий сигнал займає спектр до 15 ... 17 кГц.

Середньою потужністю P_c сигналу є потужність, яка забезпечується апаратурою під час його надходження до каналу зв'язку. На практиці частіше замість P_c користуються поняттям *динамічного діапазону* D_c, що визначається логарифмом відношення найбільшої (максимальної) миттєвої потужності сигналу ($P_{c\,max}= P_c$) до найменшої (мінімальної) $P_{c\,min}$, дозволене значення якої дорівнює потужності завад ($P_{c\,min}=P_з$)

$$D_c = \log(P_c/P_з). \qquad (1.1)$$

Ці параметри сигналу є його *обсягом*

$$V_c = T_c F_c D_c. \qquad (1.2)$$

Лінія зв'язку – фізичне середовище, яким поширюються сигнали від передавача до приймача.

Шум - це завади в каналі зв'язку.

Узагальнена схема системи передачі інформації має такий вигляд (*рис. 1.3*).

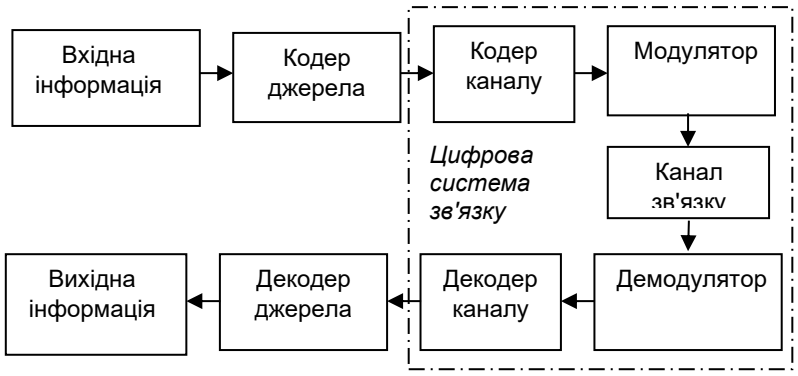


Рисунок 1.3

2. Моделі інформаційних систем.

Під *інформаційною* розуміють будь-яку систему, яка за допомогою технічних засобів виконує одну або кілька таких функцій, як збирання, передавання, перетворення, накопичення, зберігання та оброблення інформації.

За функціональною ознакою інформаційні системи можна поділити на: системи електрозв’язку; системи передачі даних; інформаційно-вимірювальні системи; системи перетворення інформації; інформаційно-пошукові системи; системи зберігання інформації; автоматизовані системи керування; системи експериментальних досліджень та ін.

Найпоширенішими в повсякденному житті є системи електрозв’язку та передачі даних, які можна об’єднати назвою *систем передачі інформації* (СПІ).

Структурна схема інформаційної системи має такий вигляд.

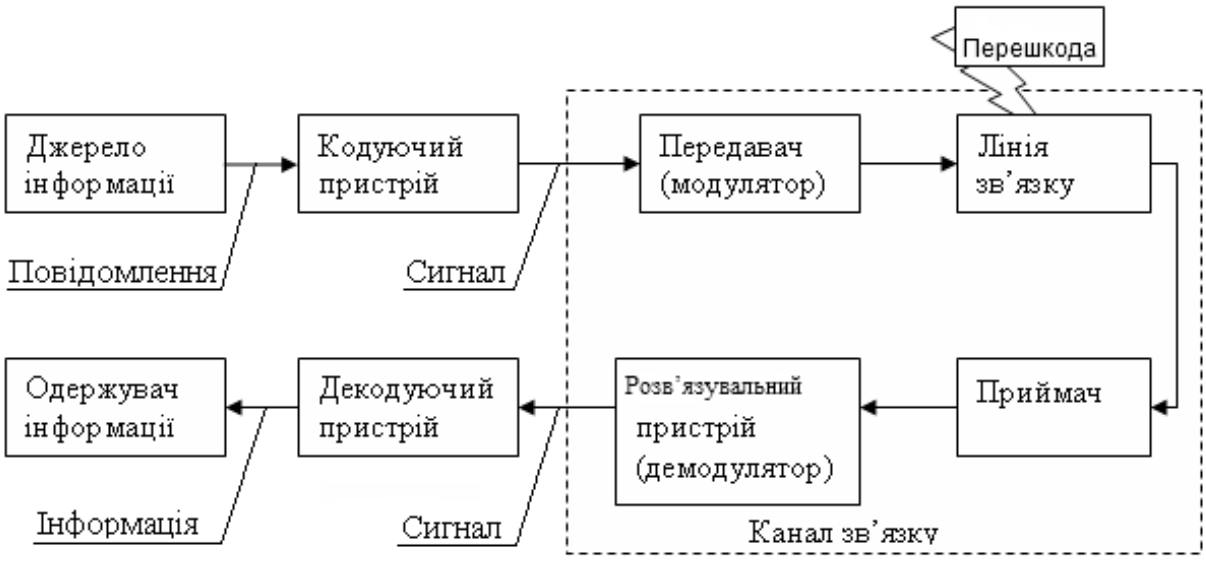


Рисунок 1.4

Призначення блоків і робота системи має такий вигляд. Кожне повідомлення для передачі його у відповідну адресу має бути попередньо перетворено у сигнал.

3. Види інформації. Теорема дискретизації

При формальному поданні знань кожному досліджуваному об'єкту ставиться у відповідність числовий код, зв'язки між об'єктами так само подаються кодами. Для переведення неформальних даних у формальний цифровий вигляд використовуються спеціальні ***таблиці кодування***. Найпростіший приклад такої таблиці - ***ASCII*** *(American Standard Code for Information Interchange)*, що зіставляє друкованим та керуючим символам числа від 0 до 127.

Інформація може бути двох видів: ***дискретна (цифрова)*** і ***неперервна (аналогова).***

Неперервна інформація – це дані, що одержані при неперервному за часом процесі змінювання деякої випадкової величини і описуються неперервними (аналоговими) функціями.

Дискретна інформація – це цифрові дані, одержані у результаті квантування (дискретизації) неперервної величини за часом, рівнем або тим і іншим одночасно (*рис.1.5*). Дискретну інформацію зберігати і обробляти набагато простіше, оскільки вона являє собою послідовність чисел. У двійковій системі числення дискретна інформація являє собою послідовність **0** та **1**.

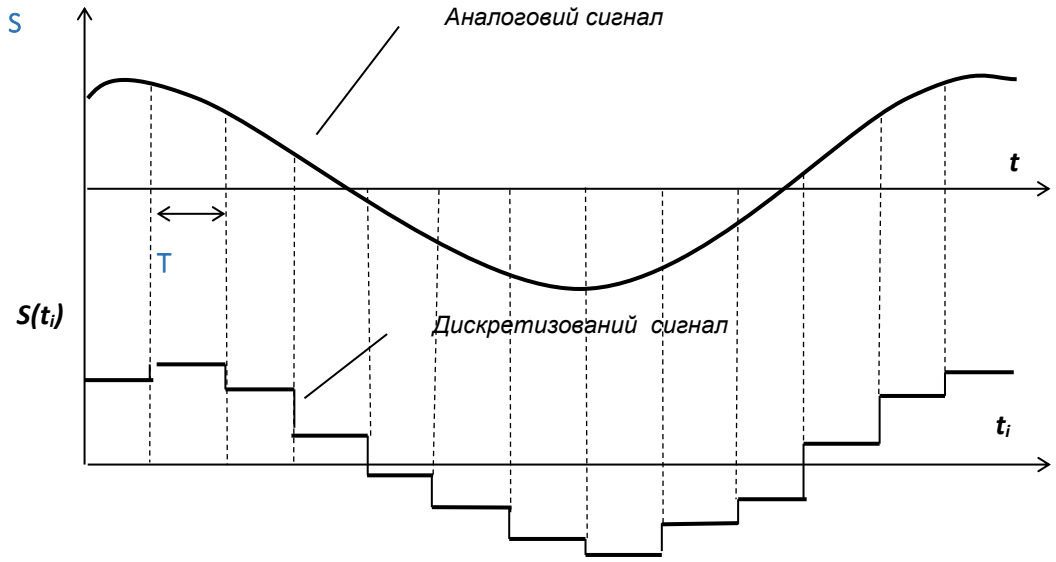


Рисунок 1. 5

За найменшу одиницю ємності цифрової інформації беруть ***біт (bit, binary digit)*** – одну позицію для двійкової цифри. Складені одиниці: 1 *Кб* = 2^{10} = 1024 *б*; 1 *Мб* = $2^{20} \approx 10^6$ *б*; 1 *Гб* = $2^{30} \approx 10^9$ *б*; 1 *Тб* = $2^{40} \approx 10^{12}$ *б*; 1 *Пб* = $2^{50} \approx 10^{15}$ *б*.

Для переведення неперервної інформації в дискретну і навпаки використовуються спеціальні пристрої модуляції/демодуляції - ***модеми***. Швидкість передачі інформації вимірюється в кількості переданих за одну секунду бітів **– *бодах (baud):* 1 *бод* = 1 *біт/с (bps)***.

Пристрій, що реалізовує процес дискретизації неперервного сигналу, називається ***аналогово-цифровим перетворювачем*** *(АЦП)*. Частота, з якою АЦП проводить виміри аналогового сигналу і видає його цифрові значення, називається ***частотою дискретизації***. Пристрій, що інтерполює дискретний сигнал у неперервний називається ***цифро-аналоговим перетворювачем***.

Чим вища частота дискретизації, тим точніше переведення неперервної інформації в дискретний сигнал. Проте із зростанням частоти зростає і розмір дискретних даних і, отже, складність їхнього оброблення, передачі і зберігання.

При всіх якісних відмінностях між неперервною і дискретною величинами існує чіткий зв'язок, встановлюваний ***теоремою дискретизації Шеннона-Котельникова.***

Як відомо з відповідного розділу математичного аналізу, будь-яка неперервна функція ***S(t)*** *може бути розкладеною на скінченному проміжку в* ***ряд Фур'є****. Суть цього розкладання полягає в тому, що функція подається у вигляді суми ряду синусоїд з різними амплітудами і*

фазами, і з кратними частотами. Коефіцієнти (амплітуди) при синусоїдах називаються ***спектром*** *функції. У гладких функцій спектр швидко спадає (із зростанням номера коефіцієнти швидко прямують до нуля). Для швидко змінюваних функцій спектр спадає поволі, оскільки в сумі гармонічного ряду таких функцій переважають синусоїди з високими частотами.*

Вважається, що сигнал має ***обмежений спектр****, якщо після певного номера всі коефіцієнти спектру прямують до нуля. Іншими словами, на заданому проміжку часу сигнал подається у вигляді скінченної суми ряду Фур'є. В цьому випадку говорять, що спектр сигналу знаходиться нижче за* ***граничну частоту*** $f_м$*, де* $f_м$ *- частота синусоїди при останньому ненульовому коефіцієнті.*

Теорема дискретизації формулюється так:

Неперервна інформація $S(t)$ *з обмеженим спектром, тобто така, що має в своєму спектрі складові з частотами, що не перевищують деяку максимальну частоту спектру* $f_м$*,, повністю відтворюється послідовністю відліків* $S(t_i)$*, узятих в дискретні моменти часу з інтервалом* $T < \frac{1}{2f_м}$.

4. Предмет теорії інформації та кодування

Теорія інформації - це розділ кібернетики, в якому за допомогою математичних методів вивчаються способи вимірювання кількості інформації, що міститься в будь-яких повідомленнях, способи кодування для економічного подання повідомлень і надійної передачі їх каналами зв'язку з завадами.

Курс теорії інформації об'єднує такі теоретичні напрями, як математичні моделі та частотний аналіз каналів і сигналів, кількісна оцінка інформації, кодування повідомлень, їх стиснення, оцінка ефективності та завадостійкості передачі кодованих повідомлень.

Одним із головних завдань теорії інформації є максимальне використання потенційних можливостей каналів зв'язку на основі оптимального кодування джерела повідомлення та його подальшого завадостійкого кодування. Це збігається з завданням **теорії кодування** - розробкою ефективних алгоритмів кодування для джерел повідомлень і передачі даних каналами зв'язку.

Теорія інформації та кодування за своєю природою дуже близька до математичних дисциплін; тому як апарат дослідження в ній застосовуються теорія скінченних полів, лінійна алгебра, комбінаторика, теорія матриць, теорія ймовірностей та математична статистика.

Без розвитку теорії інформації та кодування і впровадження її в життя практично неможливо створення складних систем керування супутниками Землі та ракетами, систем і мереж зв'язку та передачі даних, складних ЕОМ і комплексів тощо.

Методом теорії інформації є сукупність прийомів дослідження інформаційних систем (наприклад, методи оцінки інформаційної здатності джерела інформації, пропускної спроможності систем передачі інформації, інформаційної місткості пристроїв, місткості запам'ятовуючих пристроїв).

Теорія інформації і кодування встановлює критерії оцінки завадостійкості та ефективності інформаційних систем, а також указує загальні шляхи підвищення завадостійкості та ефективності інформаційних систем.

Тема 2. Кількісні характеристики джерел інформації

План

1. **Ансамблі та джерела повідомлень**
2. **Способи вимірювання інформації**
3. **Поняття про ентропія джерела. Властивості кількості інформації та ентропії**
4. **Поняття про умовну ймовірність**
5. **Модель системи передачі інформації**
6. **Види умовної ентропії**
7. **Ентропія об'єднання двох джерел інформації**

1. Ансамблі та джерела повідомлень

Матеріальному світові, що оточує людину, притаманна безліч фізичних явищ, багато з яких змінюються в часі, маючи форму фізичних процесів, тобто таких явищ, фізичні показники яких не є миттєвими, а розподіленими в часі, які можна спостерігати кожної миті.

Будь-який матеріальний об'єкт разом із спостерігачем утворює систему, яка називається *джерелом повідомлень*.

Дискретне джерело інформації – це таке джерело, яке може виробити (згенерувати) за скінчений відрізок часу тільки скінчену множину повідомлень. Кожному такому повідомленню можна співставити відповідне число, та передавати ці числа замість повідомлень.

Дискретне джерело інформації є достатньо адекватною інфор-маційною моделлю дискретних систем, а також неперервних систем, інформаційні сигнали про стан яких піддають аналого - цифровому перетворенню; таке перетворення виконується в більшості сучасних автоматизованих систем управління.

На рис. 2.1 зображено схему системи взаємозв'язаних об'єктів і спостерігачів, вкладених одне в одне, стосовно передачі відомостей про певний фізичний об'єкт певному одержувачеві. Для кожної стрілки на рис. 2.1 частина системи, розміщена ліворуч, може розглядатися, як спостережуваний об'єкт, а розташовані праворуч — як спостерігач. При цьому не має значення природа спостерігача; чи це людина, чи це якийсь прилад. Його головне завдання полягає в перетворенні відомостей про стан спостережуваного об'єкта на форму, зручну для прийняття іншими людиною або приладом.

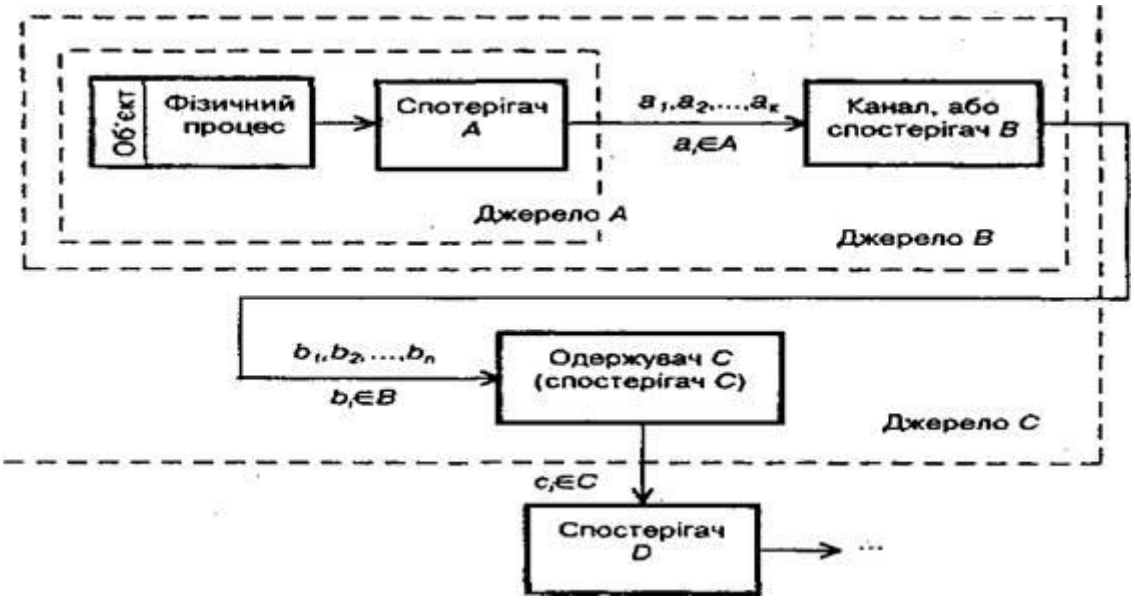


Рисунок 2.1

Стан матеріального об'єкта, а отже, і його фізичні показники можуть набувати значення з певного дискретного набору значень. Джерело повідомлень з таким об'єктом

є *дискретним.* Якщо стан матеріального об'єкта, відбитий у його фізичних показниках, набуває значення з нескінченної множини можливих значень, то таке джерело повідомлень є *неперервним.* Принципово воно може бути зведене до дискретного, якщо прийняти допустимий рівень похибки та за її допомогою з нескінченної множини можливих значень повідомлень вибрати певний дискретний набір. Саме тут у наявності похибки та її допустимому рівні криється принципова різниця між дискретним і неперервним джерелами повідомлень.

Якщо під час деякого часового проміжку дискретним джерелом вибрано деяке повідомлення a_i яке ніяк не зумовлене повідомленням a_{i-1}, вибраним у попередній проміжок часу, то таке джерело є *дискретним джерелом без пам 'яті.*

Якщо в деякому часовому проміжку дискретним джерелом вибрано повідомлення a_i, пов'язане з попереднім повідомленням a_{i-1} і статистично зумовлене ним, то таке джерело називається *дискретним джерелом із пам 'яттю.*

Крім дискретних, можуть бути також *неперервні джерела повідомлень із пам'яттю* та *без пам'яті*. Переліченими тут типами не вичерпуються всі відомі джерела повідомлень.

Якщо кожного проміжку часу дискретне джерело повідомлень вибирає одне з *n* можливих повідомлень $a_1, a_2, \dots, a_n$, то кажуть, що $A = \{a_1, a_2, \dots, a_n\}$ є дискретною множиною повідомлень, або просто множиною повідомлень А. Як правило, для Повнішого опису джерела повідомлень на множині *A* визначають її ймовірнісну міру, тобто з кожним дискретним повідомленням пов'язують ймовірність p_i його вибору джерелом. Таким чином, множині $A = \{a_1, a_2, \dots, a_n\}$ зіставляється ймовірнісна міра у вигляді множини $P = \{p_1, \dots, p_n\}$, на яку накладемо обмеження у вигляді $\sum_{i=1}^{n} p_i = 1$.

Дві множини *A* та *P* дають достатньо повний опис дискретного джерела повідомлень у вигляді його ймовірнісної моделі, і тому разом вони утворюють *ансамбль повідомлень дискретного джерела*. Це означає, що кожного проміжку часу дискретним джерелом вибирається певне повідомлення $a_i \in A$ з ймовірністю $p(a_i) = p_i \in P$. Наведене вище обмеження є природною умовою включення до складу множини *A* повної групи подій, якими виступають дискретні повідомлення. Ця вимога з'являється тут тому, що надалі треба скористатися апаратом математичної статистики, звідки й походить цей термін. Це дає змогу врахувати при розгляді всі обговорювані події-повідомлення. Зі своїм розсудом можна змінити склад можливих повідомлень в А, але слід пронормувати їх ймовірності так, щоб сума ймовірностей дорівнювала одиниці.

2. Способи вимірювання інформації

Припустимо, що стан деякого об'єкта або системи наперед відомий. Тоді повідомлення про цей стан не несе ніякої інформації для її одержувача. Якщо ж стан об'єкта змінився і джерелом передане якесь інше повідомлення про стан спостережуваного об'єкта, то це повідомлення несе нові відомості, які додадуть знання про об'єкт. Тоді можна говорити, що таке повідомлення містить деяку ***кількість інформації*** для її одержувача.

Отже, на якісному інтуїтивному рівні інформацію можна визначити як нове, наперед невідоме знання про стан деякого об'єкта або системи, а кількість інформації - кількість цього знання. Зрозуміло, що якщо нове знання збільшує загальний рівень знань про стан спостережуваного об'єкта, то кількість інформації накопичується і має адитивний характер.

До передачі джерелом повідомлення для його одержувача має місце деяка ***невизначеність*** щодо стану об'єкта спостереження. Після вибору повідомлення джерелом утворюється певна кількість інформації, що якоюсь мірою ***зменшує цю невизначеність.***

В основу теорії інформації покладено запропонований ***Клодом Шенноном*** спосіб вимірювання кількості інформації, що міститься в одній випадковій величині щодо іншої випадкової величини[1]. Цей спосіб дозволяє виразити ***кількість інформації*** числом і надає можливість об'єктивно оцінити інформацію, що міститься у повідомленні.

---

[1] ***Шеннон К.*** **Работы по теории информации и кибернетике**. - М., Изд-во иностранной литературы, 1963.

У кожному елементарному повідомленні для його одержувача міститься ***інформація*** як сукупність відомостей про стан деякого об'єкта або системи.

Для того щоб абстрагуватися від конкретного змісту інформації, тобто її смислового значення, і отримати саме загальне визначення кількості інформації, кількісну міру інформації визначають ***без урахування її смислового змісту***, а також цінності і корисності для одержувача.

До того як зв'язок відбувся, є деяка невизначеність щодо того, яке з повідомлень з можливих буде передане. Ступінь невизначеності передачі x_i можна визначити його апріорною імовірністю p_i. Отже, кількість інформації $I(X_i)$ буде деякою функцією від p_i: $I(X_i)=f(p_i)$. Визначимо вид цієї функції.

Вважатимемо, що міра кількості інформації $I(X_i)$ відповідає двом інтуїтивним властивостям:

1) якщо вибір повідомлення джерела x_i наперед відомий (немає невизначеності), тобто маємо достовірний випадок, імовірність якого $p_i=1$, то $I(X_i)=f(1)=0$;
2) якщо джерело послідовно видає повідомлення x_i і x_j, і імовірність такого вибору p_{ij} - сумісна імовірність подій x_i і x_j, то кількість інформації в цих елементарних повідомленнях дорівнює сумі кількості інформації в кожному з них.

Ймовірність сумісного випадання двох випадкових подій x_i і x_j дорівнює добутку ймовірності однієї з цих подій на ймовірність іншої за умови, що перша подія відбулася, тобто $p_{ij}=p_i \cdot p_{j/i}=P \cdot Q$.

Тоді, з *властивості 2* кількості інформації випливає, що

$$I(X_i, X_j)=I(X_i)+I(X_j)=f(P \cdot Q)=f(P)+f(Q).$$

Звідси випливає, що функції $f(p_i)$ ***логарифмічна***. Таким чином, кількість інформації зв'язана з апріорною імовірністю співвідношенням

$$I(X_i) = k \cdot \log p_i ,$$

при цьому коефіцієнт k і основа логарифма можуть бути довільними.

Для того щоб кількість інформації визначалася невід'ємним числом, взяте $k=-1$, а основу логарифма найчастіше вибирають 2, тобто

$$I(X_i) = -\log_2 p_i . \qquad (2.1)$$

У цьому випадку за одиницю кількості інформації так само береться біт. У такий спосіб ***біт*** – це кількість інформації в повідомленні дискретного джерела, алфавіт якого складається з двох альтернативних подій, які є апріорно рівноймовірними. Якщо кількість апріорно рівноймовірних подій дорівнює 2^8, то за одиницю інформації береться ***байт***.

3. Поняття про ентропія джерела. Властивості кількості інформації та ентропії

Кількість інформації, що міститься в одному елементарному повідомленні x_i, не повністю характеризує джерело. Джерело дискретних повідомлень може бути охарактеризовано середньою кількістю інформації, що припадає на одне елементарне повідомлення, і називається ***ентропією джерела***, тобто питомою кількістю інформації

$$H(X) = -\sum_{i=1}^{k} p_i \log_2 p_i , \quad i=1\ldots k , \qquad (2.2)$$

де k - об'єм алфавіту джерела.

Фізичний зміст ентропії - це середньостатистична міра невизначеності знань одержувача інформації щодо стану спостережуваного об'єкта.

У формулі (2.2) статистичне усереднювання (тобто обчислювання математичного сподівання випадкової величини) виконується за всім ансамблем повідомлень джерела. При цьому необхідно враховувати всі імовірнісні зв'язки між різними повідомленнями. Чим вища ентропія джерела, тим більша кількість інформації в середньому закладається в кожне повідомлення, тим важче її запам'ятати (записати) або передати каналом зв'язку.

Необхідні втрати енергії на передачу повідомлення пропорційні ентропії (середній кількості інформації на одне повідомлення). Звідси випливає, що кількість інформації в послідовності з N повідомлень визначається кількістю цих повідомлень і ентропією джерела, тобто $I(N)=N\cdot H(X)$.

Ентропія як кількісна міра інформаційності джерела має такі ***властивості***:

1) *ентропія дорівнює нулю, якщо хоча б одне з повідомлень достовірне*;

2) *ентропія завжди більша або дорівнює нулю, є величиною дійсною і обмеженою*;

3) *ентропія джерела з двома альтернативними подіями може змінюватися від 0 до 1*;

4) *ентропія - величина адитивна: ентропія джерела, повідомлення якого складаються з повідомлень декількох статистично незалежних джерел, дорівнює сумі ентропій цих джерел*;

5) *ентропія максимальна, якщо всі повідомлення мають однакову імовірність. Таким чином,*

$$H_{\max}(X) = \log_2 k \,. \tag{2.3}$$

Вираз (2.3) називається формулою ***Хартлі***. ЇЇ легко вивести з формули *Шеннона* (2.2), припустивши, що $p_i=1/k$, де $i=1\ldots k$.

Розглянемо дискретні випадкові величини (д. в. в.) X і Y, що задані законами розподілів їхніх ймовірностей $P(X=X_i)=p_i$, $P(Y=Y_i)=q_j$ та розподілом сумісних ймовірностей системи д. в. в. $P(X=X_i,\ Y=Y_j)=p_{ij}$. Тоді ***кількість інформації, що міститься в д. в. в. X щодо д. в. в. Y – взаємна інформація***, визначається так:

$$I(X,Y) = \sum_{ij} p_{ij} \log_2 \frac{p_{ij}}{p_i q_j}$$
$$H(X) = I(X,X) \,. \tag{2.4}$$

Наведемо ***властивості кількості інформації*** і ***ентропії***:

1) $I(X, Y)\geq 0$; $I(X, Y)=0 \Leftrightarrow$ ***X і Y незалежні*** (*одна в. в. нічим не описує іншу*);

2) $I(X, Y)=I(Y, X)$;

3) $HX=0 \Leftrightarrow X=const$;

4) $I(X, Y)=HX+HY-H(X, Y)$, *де* $H(X,Y) = -\sum_{ij} p_{ij} \log_2 p_{ij}$;

5) $I(X, Y) \leq I(X, X)$; *якщо* $I(X, Y)=I(X, X) \Rightarrow X=f(Y)$.

4. Поняття про умовну ймовірність

Раніше отримана формула ентропії (2.4) визначає її середньою кількістю інформації, що припадає на одне повідомлення джерела ***статистично незалежних повідомлень.*** Така ентропія називається ***безумовною***.

Як відомо з відповідного розділу математичної статистики, мірою порушення статистичної незалежності повідомлень x і y є ***умовна ймовірність*** $p(x/y)$ появи повідомлення x_i за умови, що вже вибрано повідомлення y_j або умовна ймовірність появи повідомлення y_j, якщо вже отримане повідомлення x_i, причому в загальному випадку $p(x/y)\neq p(y/x)$.

Умовну ймовірність можна отримати з безумовної ймовірності $p(x)$ чи $p(y)$ та сумісної ймовірності системи в. в. $p(x, y)$ за формулою множення ймовірностей:

$$p(x, y)=p(x)\cdot p(y/x), \tag{3.1}$$

$$p(x, y)=p(y)\cdot p(y/x), \qquad (3.2)$$

звідси

$$p(y/x) = p(x,y)/p(x),$$
$$p(x/y) = p(x,y)/p(y).$$

В окремому випадку для статистично незалежних повідомлень маємо: $p(y/x)=p(y)$, $p(x/y)=p(x)$.

При ***існуванні статистичної залежності*** між повідомленнями джерела факт вибору одного з повідомлень зменшує або збільшує ймовірності вибору інших повідомлень до ***умовних ймовірностей***. Відповідно змінюється й кількість інформації, що міститься в кожному з цих повідомлень, згідно з (1.2). Ентропія такого джерела також змінюється відповідним чином, причому обчислюється ентропія за тією самою формулою (1.3), але вже з ***урахуванням умовних ймовірностей***. Така ентропія називається ***умовною***.

5. Модель системи передачі інформації

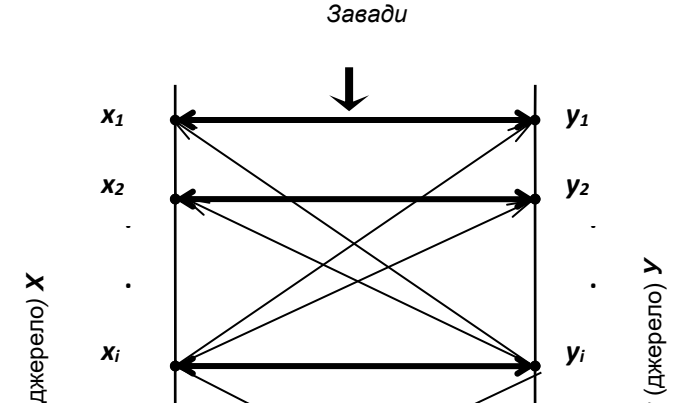
Розглянемо модель системи спостереження, перетворення, збору і зберігання інформації, що складається з двох джерел, між повідомленнями яких існує статистичний взаємозв'язок. Нехай джерело X задано моделлю - ***ансамблем повідомлень*** $\{x_1, x_2, \ldots, x_i, \ldots, x_k\}$ і ***рядом розподілу*** $P(X)$ їхніх ймовірностей, а джерело Y - ансамблем $\{y_1, y_2, \ldots, y_j, \ldots, y_l\}$ і розподілом $P(Y)$.

Ніяких обмежень на алфавіти X і Y не накладається. Вони можуть навіть збігатися $(X=Y)$. Тоді можна аналізувати і враховувати взаємозв'язок між повідомленнями одного джерела, що рознесені за часом. Найбільш поширеною такою моделлю є послідовності елементарних повідомлень $\{x_i\}$, умовна ймовірність $p(x_i/x_{i-1})$ кожного з яких залежить тільки від попереднього значення x_{i-1} за умови появи всіх $i-1$ повідомлень. Такі послідовності називають ***ланцюгами Маркова***.

Алфавіти X і Y можуть і не збігатися ($X\neq Y$), хоча між їхніми елементами може бути встановлена взаємна відповідність. Джерело X описується моделлю - ансамблем повідомлень $\{x_i\}$ і рядом розподілу $P(X)$. У той самий час джерело X може виступати як об'єкт спостереження для одержувача інформації Y і разом з ним утворювати нове джерело, яке описується моделлю - ансамблем $\{y_j\}$ і розподілом $P(Y)$. Між джерелом X і спостерігачем Y існує ***канал зв'язку***, на який впливають ***завади***, що можуть порушити процес вибору спостерігачем Y повідомлень алфавіту $y_j \in Y$, що, у свою чергу, порушує відповідність між повідомленнями $x_i \in X$ і $y_j \in Y$.

Алфавіти X і Y можуть бути однакового $(k=l)$ і неоднакового ($k\neq l$) об'ємів. Звичайно розглядаються ситуації, коли $k=l$ або $k<l$. Система спостереження при $k=l$ має природне пояснення. Спостерігач Y повинен реагувати повідомленнями y_j $(j=1, \ldots, l)$ на кожний стан джерела X, представлений повідомленням x_i $(i=1,\ldots,k)$, при чому кожному повідомленню x_i джерела X відповідає повідомлення y_j з Y: $x_1 \to y_1$, $x_2 \to y_2$, …, $x_i \to y_i$, …, $x_k \to y_k$.

Дана модель показана на *рис.1.3*, за винятком елемента y_l з Y, де $l=k+1$ (жирними лініями показані напрями взаємооднозначної відповідності $X \Leftrightarrow Y$).



У разі, коли джерелом X вибране деяке повідомлення x_i, якому повинне відповідати повідомлення y_j при $i=j$, то через вплив завад джерелом Y може бути вибране *будь-яке* з повідомлень y_j, $j=1\ldots k$ з ймовірністю $p(y_j/x_i)$, причому умовна імовірність правильного вибору повідомлення $p(y_i/x_i)$. Інші повідомлення y_j, що визначаються умовними ймовірностями $p(y_j/x_i)$, де $i\neq j$, є помилковими. Така модель дозволяє досліджувати систему передачі інформації з боку спостерігача Y.

Водночас таку систему можна досліджувати з позиції об'єкта спостереження X. Для цього потрібно знати факт вибору джерелом Y деякого повідомлення y_j. Після цього можна з умовною імовірністю $p(x_i/y_j)$ говорити, що об'єкт X знаходиться у стані x_i. Це твердження правильне при $i=j$ або неправильне при $i\neq j$.

Існують ситуації, коли систему ускладнюють, вибираючи $l>k$ (частіше $l=k+1$, *див. рис. 1.1*). При цьому повідомлення y_l не відповідає ніякому з повідомлень x_i і є ознакою особливого стану спостерігача Y - *стирання повідомлення*.

Взагалі статистична залежність джерела Y від джерела X задається ***матрицею прямих переходів*** повідомлень x_i ($i=1\ldots k$) джерела X в повідомлення y_j ($j=1\ldots k$) джерела Y:

$P(Y/X)=$

X	Y					
	Y_1	Y_2		y_i		y_k
x_1	$p(y_1/x_1)$	$p(y_2/x_1)$		$p\ (y_j/x_1)$		$p(y_k/x_1)$
x_2	$p(y_1/x_2)$	$p(y_2/x_2)$		$p(y_j/x_2)$		$p(y_k/x_2)$
…	…	…		…		…
x_i	$p(y_1/x_i)$	$p(y_2/x_i)$		$p(y_j/x_i)$		$p(y_k/x_i)$
…	…	…		…		…
x_k	$p(y_1/x_k)$	$p(y_2/x_k)$		$p(y_j/x_k)$		$p(y_k/x_k)$

На головній діагоналі цієї матриці розташовані умовні ймовірності прямої відповідності типу $x_1\rightarrow y_1$, $x_2\rightarrow y_2$, …, $x_k\rightarrow y_k$, які характеризують правильний вибір джерелом Y повідомлень (тобто відповідно до повідомлень джерела X).

Матриця (1.9) відображає вплив завад у каналі зв'язку між джерелом X і спостерігачем Y (*рис.1.1*). Якщо завади неістотні або відсутні, то маємо однозначну відповідність $x_i\rightarrow y_j$ з умовної імовірності $p(y_i/x_i)=1$ для $i=1\ldots k$, решта ймовірностей $p(y_i/x_i)=0$ для всіх $i\neq j$.

Кожний рядок в (1.9) є спотвореним розподілом ймовірностей $p(y_j/x_i)$ появи повідомлень $y_j\in Y$, причому для кожного рядка повинна виконуватися умова нормування

$$\sum_j p(y_j / x_i) = 1, \quad i = 1\ldots k \qquad (3.3)$$

Статистична залежність джерела X від джерела Y подається ***матрицею зворотних переходів*** типу $x_i\leftarrow y_j$ з умовних ймовірностей $p(x_i/y_j)$:

$P(X/Y)=$

X	Y					
	y_1	Y_2	…	y_i		y_k
x_1	$p(x_1/y_1)$	$p(x_1/y_2)$	…	$p(x_1/y_j)$		$p(x_1/y_k)$
x_2	$p(x_2/y_1)$	$p(x_2/y_2)$	…	$p(x_2/y_j)$		$p(x_2/y_k)$
…			…	…		…
x_i	$p(x_i/y_1)$	$p(x_i/y_2)$	…	$p(x_i/y_j)$		$p(x_i/y_k)$
…		…	…	…		…
x_k	$p(x_k/y_1)$	$p(x_k/y_2)$	…	$p(x_k/y_j)$		$p(x_k/y_k)$

Матриця (1.11) складається з k розміщених стовпцями варіантів первинних розподілів ймовірностей ансамблю X, що на собі відчуває статистичний вплив повідомлень y_j джерела Y. Для кожного такого розподілу виконується умова нормування

$$\sum_i p(x_i / y_j) = 1, \quad j = 1 \ldots k \quad . \tag{3.4}$$

Отже, якщо задані ансамбль X і матриця прямих переходів (1.9), то, використовуючи безумовні імовірності $P(X)=\{p(x_i)\}$, за формулою (1.7) можна знайти матрицю сумісних ймовірностей

$$p(x_i, y_j) = \begin{bmatrix} p(x_1, y_1)\ p(x_1, y_2) \ldots p(x_1, y_k) \\ p(x_2, y_1)\ p(x_2, y_2) \ldots p(x_2, y_k) \\ \ldots \quad \ldots \quad \ldots \quad \ldots \\ p(x_k, y_1)\ p(x_k, y_2) \ldots p(x_k, y_k) \end{bmatrix}. \tag{3.5}$$

Виконавши у (3.5) згортку за i, дістанемо ряд розподілу безумовних ймовірностей $P(Y)=\{p(y_j)\}$, $j=1\ldots k$:

$$p(y_j) = \sum_{i=1}^{k} p(x_i, y_j), \quad j = 1 \ldots k \quad , \tag{3.6}$$

а виконавши згортку за j, - розподіл $P(X)=\{p(x_i)\}$, $i=1\ldots k$:

$$p(x_i) = \sum_{j=1}^{k} p(x_i, y_j), \quad i = 1 \ldots k \, . \tag{3.7}$$

6. Види умовної ентропії

Вирізняють ***часткову*** та ***загальну умовні ентропії*** джерела повідомлень.

Часткова умовна ентропія - це кількість інформації, що припадає на одне повідомлення джерела X за умови встановлення факту вибору джерелом Y повідомлення y_j, або кількість інформації, що припадає на одне повідомлення джерела Y за умови, що відомий стан джерела X:

$$H(X / y_j) = -\sum_i p(x_i / y_j) \log_2 p(x_i / y_j), \; j = 1 \ldots l \quad , \tag{3.8}$$

$$H(Y / x_i) = -\sum_j p(y_j / x_i) \log_2 p(y_j / x_i), \; i = 1 \ldots k \quad , \tag{3.9}$$

де $X = \{x_1, x_2, \ldots, x_i, \ldots, x_k\}$, $Y = \{y_1, y_2, \ldots, y_j, \ldots, y_l\}$ - алфавіти повідомлень; x_i - певне повідомлення джерела X, щодо якого визначається часткова умовна ентропія $H(Y/x_i)$ алфавіту Y за умови вибору джерелом X повідомлення x_i; y_j - певне повідомлення джерела Y, щодо якого визначається часткова умовна ентропія $H(X/y_j)$ алфавіту X за умови вибору повідомлення y_j; i - номер повідомлення з алфавіту X; j - номер повідомлення з алфавіту Y; $p(x_i/y_j)$, $p(y_j/x_i)$ – умовні імовірності.

Загальна умовна ентропія визначається так:

$$H(X/Y) = \sum_j p(y_j) H(X/y_j), \tag{3.10}$$

$$H(Y/X) = \sum_i p(x_i) H(Y/x_i). \tag{3.11}$$

Отже, загальна умовна ентропія (3.10) - це середньостатистична кількість інформації (математичне сподівання), що припадає на будь-яке повідомлення джерела X, якщо відомий його статистичний взаємозв'язок з джерелом Y. Так само загальна умовна ентропія (1.19) - це середня кількість інформації, яка міститься в повідомленнях джерела Y за наявності статистичного взаємозв'язку з джерелом X.

З урахуванням (1.16), (1.17) та (1.7), (1.8) вирази (1.18), (1.19) набувають такого вигляду:

$$H(X/Y) = -\sum_j p(y_j) \sum_i p(x_i/y_j) \log_2 p(x_i/y_j) = \\ = -\sum_j \sum_i p(x_i, y_j) \log_2 p(x_i/y_j), \tag{3.12}$$

$$H(Y/X) = -\sum_i p(x_i) \sum_j p(y_j/x_i) \log_2 p(y_j/x_i) = \\ = -\sum_i \sum_j p(x_i, y_j) \log_2 p(y_j/x_i), \tag{3.13}$$

де $p(x_i, y_j)$ - сумісна імовірність появи повідомлень x_i, y_j; $p(x_i/y_j)$, $p(y_j/x_i)$ – їх умовні імовірності.

Властивості умовної ентропії:

1) *якщо джерела повідомлень X і Y статистично незалежні, то умовна ентропія джерела X стосовно Y дорівнює безумовній ентропії джерела X і навпаки*:
$H(X/Y)=H(X)$, $H(Y/X)=H(Y)$;

2) *якщо джерела повідомлень X і Y настільки статистично взаємозв'язані, що виникнення одного з повідомлень спричиняє безумовну появу іншого, то їхні умовні ентропії дорівнюють нулю*:
$H(X/Y)=H(Y/X)=0$;

3) *ентропія джерела статистично взаємозалежних повідомлень (умовна ентропія) менша від ентропії джерела незалежних повідомлень (безумовної ентропії)*:
$H(X/Y)<H(X)$, $H(Y/X)<H(Y)$.

З *властивості 3* випливає поняття ***статистичної надмірності***, *обумовленої наявністю статистичної залежності між елементами повідомлення*:

$$\rho_{X/Y} = 1 - H(X/Y)/H(X), \tag{3.14}$$

де $H(X/Y)$ - загальна умовна ентропія джерела X стосовно джерела Y; $H(X)$ - безумовна ентропія джерела X.

З урахуванням виразу (1.5) загальна статистична надлишковість алфавіту джерела інформації визначається так:

$$\rho_{X,Y} = \rho_X + \rho_{X/Y} - \rho_X \rho_{X/Y}. \quad (3.15)$$

У разі малих значень ρ_X, $\rho_{X/Y}$ статистична надлишковість визначається виразом

$$\rho_{X,Y} = \rho_X + \rho_{X/Y}. \quad (3.16)$$

7. Ентропія об'єднання двох джерел інформації

Ентропію ***H(X, Y)*** об'єднання двох джерел інформації ***X*** і ***Y*** знаходять через імовірності ***p(xi, yj)*** системи випадкових повідомлень ***xi, yj*** для всіх ***i=1...k,*** і ***j=1...l***. Для цього складається матриця ймовірностей системи двох статистично залежних джерел

$$p(x_i, y_j) = \begin{bmatrix} p(x_1, y_1)\ p(x_1, y_2) \dots p(x_1, y_l) \\ p(x_2, y_1)\ p(x_2, y_2) \dots p(x_2, y_l) \\ \dots \quad \dots \quad \dots \quad \dots \\ p(x_k, y_1)\ p(x_k, y_2)\ \dots p(x_k, y_l) \end{bmatrix}. \quad (3.17)$$

Ентропія об'єднання двох джерел H(X, Y) (*взаємна ентропія*) - це середня кількість інформації, що припадає на два будь-які повідомлення джерел ***X*** і ***Y***:

$$H(X,Y) = -\sum_{ij} p(x_i, y_j) \log_2 p(x_i, y_j). \quad (3.18)$$

З рівності $p(x_i, y_j) = p(y_j, x_i)$ випливає, що

$$H(X, Y) = H(Y, X).$$

Скориставшись формулами (1.7), (1.12), запишемо вираз (1.26) так:

$$H(X,Y) = -\sum_{ij} p(x_i) p(y_j / x_i) \log_2 \left[p(x_i) p(y_j / x_i) \right] =$$

$$= -\sum_{ij} p(x_i) p(y_j / x_i) \log_2 p(x_i) - \sum_{ij} p(x_i) p(y_j / x_i) \log_2 p(y_j / x_i) =$$

$$= -\left\{ \sum_i p(x_i) \log_2 p(x_i) + \sum_i p(x_i) \sum_j p(y_j / x_i) \log_2 p(y_j / x_i) \right\}.$$

Перший доданок цього виразу відповідає безумовній ентропії ***H(X)*** (1.3), а другий - умовній ***H(Y/X)*** (1.21). Звідси

$$H(X, Y) = H(X) + H(Y/X). \quad (3.19)$$

З симетричності формул (1.7) – (1.8) випливає, що аналогічний вираз для взаємної ентропії ***H(X, Y)*** має вигляд

$$H(X, Y) = H(Y) + H(X/Y), \quad (3.20)$$

звідси

$$H(Y/X) = H(X,Y) - H(X), \quad (3.21)$$

$$H(X/Y) = H(X,Y) - H(Y). \quad (3.22)$$

Кількість інформації, що припадає на одне повідомлення, передане по каналу зв'язку джерелом ***X*** спостерігачу ***Y*** (*рис.1.3*), за наявності завад і статистичного взаємозв'язку

ансамблів $\boldsymbol{X}$ і $\boldsymbol{Y}$ з урахуванням виразів (1.27), (1.28) і *властивості 4* кількості інформації і ентропії знаходиться за формулою

$$\boldsymbol{I(X,Y)=H(Y)+H(X)-H(X,Y)=H(X)-H(X/Y)=H(Y)-H(Y/X)}. \qquad (3.23)$$

Властивості ентропії об'єднання двох джерел інформації:

1) *при статистичній незалежності джерел* $\boldsymbol{X}$ *і* $\boldsymbol{Y}$ *їх взаємна ентропія дорівнює сумі ентропій кожного з джерел*, *тобто* $\boldsymbol{H(X, Y)=H(X)+H(Y)}$;
2) *при повній статистичній залежності джерел* $\boldsymbol{X}$ *і* $\boldsymbol{Y}$ *їх взаємна ентропія дорівнює безумовній ентропії одного з джерел, тобто* $\boldsymbol{H(X,Y)=H(X)=H(Y)}$;
3) *взаємна ентропія статистично залежних джерел* $\boldsymbol{X}$ *і* $\boldsymbol{Y}$ *менша суми безумовних ентропій кожного з них, тобто* $\boldsymbol{H(X, Y) \le H(X)+H(Y)}$.

Тема 3. Характеристики дискретних і неперервних джерел інформації

План

1. **Продуктивність дискретного джерела інформації. Швидкість передачі інформації**
2. **Інформаційні втрати при передачі інформації по дискретному каналу зв'язку**
3. **Пропускна здатність дискретного каналу. Основна теорема про кодування дискретного джерела**
4. **Математичні моделі джерел неперервних повідомлень та їх статистичні Характеристики**
5. **Інформаційні характеристики джерел неперервних повідомлень.**

1. Продуктивність дискретного джерела інформації. Швидкість передачі інформації

Нехай дискретне джерело X видає послідовність повідомлень $\{x_i\}$, заданих рядом ймовірностей $\{p_i\}$.

Якщо джерелом вибирається одне повідомлення x_i, то ним виробляється певна кількість інформації. Тоді швидкість утворення джерелом інформації повідомлень - ***продуктивність джерела*** щодо конкретного повідомлення можна визначити так:

$$V_{дж.i} = \frac{I(X_i)}{\tau_i}, \tag{5.1}$$

де через τ_i позначено проміжок часу вибору повідомлення x_i.

Оскільки джерелом за деякий часовий інтервал вибирається велика кількість повідомлень і в загальному випадку $\tau_i \neq \tau_j$, то продуктивність джерела інформації прийнято характеризувати середнім значенням

$$V_{дж} = \frac{1}{\tau_{сер}} \sum_{i=1}^{k} p_i I(X_i) = \frac{H(X)}{\tau_{ср}}, \tag{5.2}$$

де $\tau_{сер}$ – середній час вибору джерелом одного повідомлення.

Отже, ***продуктивність джерела інформації визначається середньою кількістю інформації, що виробляється джерелом за одиницю часу***.

Повідомлення x_i передається по каналу зв'язку спостерігачеві Y, роль якого відіграє приймальний пристрій. Вибір повідомлень $y_j \in Y$ джерелом Y характеризує процес передачі інформації по каналу зв'язку від джерела X на вихід джерела Y. При цьому ***взаємна кількість інформації I(X, Y) - це середня кількість інформації про стан джерела X, що міститься в одному повідомленні джерела Y***.

Оскільки на вибір кожного повідомлення y_j джерелом Y витрачається час τ, то ***швидкість передачі інформації по каналу зв'язку*** знаходиться за формулою

$$V = \frac{I(X,Y)}{\tau}. \tag{5.3}$$

2. Інформаційні втрати при передачі інформації по дискретному каналу зв'язку

.Математично канал дискретної інформації описується ансамблем повідомлень на вході $\{x_i\}$, $\{p_i\}$ та йому відповідними йому значеннями на виході $\{y_j\}$, а також набором умовних ймовірностей $p(y_j/x_i)$ вибору сигналу y_j на виході при передачі сигналу x_i.

Задача каналу зв'язку полягає в тому, щоб отримати однозначну відповідність повідомлення y_i повідомленню x_i, тобто повинна виконуватися умова $p(y_i/x_i)=1$ при $i=j$ і $p(y_j/x_i)=0$ при $i \neq j$. У цьому випадку канал зв'язку називається ***каналом без шуму***.

Виконання умов використання ***каналу без шуму*** означає ***повний збіг ансамблів X*** і ***Y***, тобто ***повний статистичний взаємозв'язок джерел***. Звідси випливає, що

$$H(X/Y)=H(Y/X)=0. \tag{5.4}$$

Тоді середня кількість інформації на одне повідомлення джерела ***X***, яка дорівнює ентропії ***H(X)***, при повній відсутності інформаційних втрат відповідає такій самій кількості прийнятої інформації ***H(Y),*** тобто

$$I(X,Y)=H(X)=H(Y)=H(X,Y). \quad (5.5)$$

Отже, ***при відсутності завад кількість переданої інформації дорівнює ентропії об'єднання двох джерел або безумовної ентропії одного з них***.

При ***високому рівні завад*** спостерігається ***повна статистична незалежність джерел*** ***X*** і ***Y***, тобто

$$H(X/Y)=H(X), \quad (5.6)$$

$$H(Y/X)=H(Y), \quad (5.7)$$

$$H(X,Y)= H(X)+H(Y). \quad (5.8)$$

У даному випадку через сильний вплив завад порушується взаємозв'язок джерел, і інформація від джерела ***X*** джерелу ***Y*** не передається, отже,

$$I(X, Y)= 0. \quad (5.9)$$

У проміжному випадку ***неабсолютного статистичного взаємозв'язку джерел X, Y*** завади деякою мірою спотворюють передані повідомлення. При цьому умовна ентропія змінюється в межах від нуля (при повній статистичній залежності джерел) до безумовної ентропії (за відсутності статистичної залежності джерел), тобто

$$0 \le H(X/Y) \le H(X), \quad (5.10)$$

$$0 \le H(Y/X) \le H(Y). \quad (5.11)$$

Кількість інформації, що передається джерелом X спостерігачу Y, можна визначити так. Якщо джерелом ***X*** вибрано повідомлення $x_i \in X$, то ним, в середньому, передається кількість інформації ***H(X)***. Джерело ***Y***, вибравши повідомлення $y_j \in Y$, за умови порушення повної статистичного залежності джерел ***X*** і ***Y*** виробляє певну кількість інформації ***H(X/Y)***.

Після вибору повідомлення $y_j \in Y$ джерелом ***Y*** приймається рішення щодо того, яке з повідомлень $x_i \in X$ передане. Прийнявши це рішення, джерело ***Y*** виробляє кількість інформації про стан джерела ***X***, яка дорівнює ***HX***. Проте до цього джерело ***Y*** вже має ***H(X/Y)*** бітів інформації про ***X***, тому кількість переданої по каналу зв'язку інформації як кількість нового відсутнього знання визначається різницею ***HX*** і ***H(X/Y)***:

$$I(X, Y)=H(X)-H(X/Y). \quad (5.12)$$

Вираз (5.12) за відсутності завад збігається з виразом (5.5), а при високому рівні завад, тобто при виконанні умови статистичної незалежності джерел (5.6 – 5.8) - з виразом (5.9).

Отже, ***інформаційні втрати в каналі визначаються умовною ентропією одного джерела щодо іншого, а кількість переданої інформації - безумовною ентропією джерела і інформаційними втратами*** за формулою (5.12).

З властивості симетричності взаємної ентропії випливає рівність

$$H(X)+H(Y/X)=H(Y)+H(X/Y). \quad (5.13)$$

Віднявши від обох частин цієї рівності суму ***H(X/Y)+H(Y/X),*** дістанемо

$$H(X)-H(X/Y)=H(Y)-H(Y/X). \quad (5.14)$$

Звідси випливає властивість симетричності взаємної інформації

$$I(X, Y)=I(Y, X). \quad (5.15)$$

Скориставшись виразами (5.12), (5.13), (5.15), маємо

$$I(X,Y)=HX-H(X/Y)=HX-(H(X,Y)-HY)=HX+HY-H(X,Y), \quad (5.16)$$

$$I(Y,X)=HY-H(Y/X)=HY-(H(Y,X)-HX)=HY+HX-H(Y,X), \quad (5.17)$$

чим доводяться *властивість 4* кількості інформації і повна симетричність виразів (5.16), (5.17).

2. Пропускна здатність дискретного каналу. Основна теорема про кодування дискретного джерела.

Максимально можлива швидкість передачі інформації по каналу називається ***пропускною здатністю***, або ***ємністю каналу зв'язку C***.

Виходячи з виразів (5.3) і (5.12), дістанемо формулу

$$C = \frac{1}{\tau}\left[I(X,Y)\right]_{\max} = \frac{1}{\tau}\left[H(X) - H(X/Y)\right]_{\max}. \quad (5.18)$$

Очевидно, що вираз (5.18) досягає максимуму при абсолютній статистичній залежності джерел X, Y, тобто за відсутності або при малому рівні завад. У цьому випадку $H(X/Y)=0$, і оскільки ентропія максимальна у разі рівноімовірних повідомлень, то формула (5.18) набуває вигляду:

$$C = \frac{1}{\tau} H(X)_{\max} = \frac{1}{\tau}\log_2 k. \quad (5.19)$$

Вираз (5.19) визначає пропускну здатність за відсутності завад.

У разі, коли в каналі наявні завади, умовна ентропія на його вході і виході $H(X/Y)$ знаходиться в діапазоні $0 \le H(X/Y) \le H(X)$. Тоді пропускна здатність каналу визначається за формулою

$$C = \frac{1}{\tau}\left[\log_2 k - H(X/Y)\right]. \quad (5.20)$$

При зменшенні рівня завад пропускна здатність каналу C прямує до максимального значення (5.19), а при збільшенні рівня завад – до нуля.

Основна теорема кодування дискретного джерела, сформульована і доведена *К. Шенноном*, полягає в такому.

Припустимо, що при передачі інформації використовується канал без шуму. Розглянемо безнадмірні (рівноймовірні) вхідні повідомлення, що характеризуються максимальною ентропією $H(X)_{max}$. У цьому випадку може бути досягнута максимальна швидкість передачі в каналі

$$C = V \cdot H(X)_{\max} = V \log_2 k, \quad (5.21)$$

де $V=1/T$; T - тривалість передачі одного елементарного повідомлення (символу) x_i; $\log_2 k$ - максимальна ентропія джерела з алфавітом об'ємом k.

Якщо статистична надлишковість джерела інформації більше нуля, то швидкість передачі інформації по каналу

$$V < \frac{C}{H(X)}. \qquad (5.22)$$

Як доведено *К. Шенноном*, *при будь-якій статистичній надмірності джерела інформації існує такий спосіб кодування повідомлень, при якому може бути досягнута швидкість передачі інформації по каналу без шуму, скільки завгодно близька до його пропускної здатності.* Таким чином, *умовою узгодженості джерела інформації і каналу передачі є відповідність продуктивності першого пропускній здатності другого.*

Теорема Шеннона про кодування дискретного джерела за відсутності завад[1] стверджує про таке.

Якщо пропускна здатність каналу без шуму перевищує швидкість створення джерелом повідомлень - його продуктивність, тобто

$$V \log_2 k \geq V_{дж} H(X),$$

то існує спосіб кодування/ декодування повідомлень джерела з ентропією H(X), що забезпечує скільки завгодно високу надійність зіставлення прийнятих кодових комбінацій переданим, інакше - такого способу немає.

За ***наявності завад*** в каналі ***основна теорема кодування*** узагальнюється такою теоремою:

Якщо для будь-якого повідомлення дискретного джерела X задана ймовірність його спотворення в каналі ε, то для будь-якого ε > 0 існує спосіб передачі інформації зі швидкістю

$$V' < \frac{C}{H(X)},$$

скільки завгодно близькою до

$$V = \frac{C}{H(X)},$$

при якому ймовірність помилки в каналі буде менше ε. Цей спосіб утворює завадостійкий код.

Фано доведена ***зворотна теорема кодування джерела за наявності завад***:

Якщо швидкість передачі інформації по каналу зв'язку з шумом $V' > \frac{C}{H(X)}$, ***то можна знайти таке ε > 0, що ймовірність помилки при передачі повідомлення при будь-якому методі кодування/ декодування буде не менше ε (очевидно ε зростає із зростанням V').***

Приклад 1. Матриця сумісних ймовірностей каналу зв'язку має вигляд

$$p(x_i, y_j) = \begin{bmatrix} 0{,}15 & 0{,}15 & 0 \\ 0 & 0{,}25 & 0{,}1 \\ 0 & 0{,}2 & 0{,}15 \end{bmatrix}.$$

Знайти інформаційні втрати, пропускну здатність і швидкість передачі інформації по дискретному каналу зв'язку, якщо час передачі одного повідомлення $\tau = 10^{-3}$ *с*.

Розв'язання

Інформаційні втрати в каналі зв'язку визначаються умовною ентропією $H(X/Y)$ одного джерела щодо іншого.

Для того щоб обчислити повну умовну ентропію $H(X/Y)$, потрібно знайти розподіли безумовних ймовірностей $p(x_i)$, $p(y_j)$ і побудувати матрицю умовних ймовірностей $p(x_i/y_j)$.

Безумовний закон розподілу $p(x_i)$ знаходимо, виконавши в матриці сумісних ймовірностей $p(x_i, y_j)$ згортку за j:

$$p(x_1)= 0{,}15+0{,}15+0=0{,}3,\ \ i=1;$$
$$p(x_2)= 0+0{,}25+0{,}1=0{,}35,\ \ i=2;$$
$$p(x_3)= 0+0{,}2+0{,}15=0{,}35,\ \ i=3.$$

Перевіряємо умову нормування

$$p(x_1)+ p(x_2)+ p(x_3)=0{,}3+0{,}35+0{,}35=1.$$

Виходячи з розподілу безумовних ймовірностей д. в. в. X, обчислимо її ентропію:

$$HX = -\left(0{,}3\log_2 0{,}3 + 0{,}35\log_2 0{,}35 + 0{,}35\log_2 0{,}35\right) \approx 1{,}581\ (\textit{біт/сим}).$$

Безумовний закон розподілу $p(y_j)$ знаходимо, виконавши в матриці сумісних ймовірностей $p(x_i, y_j)$ згортку за i:

$$p(y_1)= 0{,}15+0+0=0{,}15,\ \ j=1;$$
$$p(y_2)= 0{,}15+0{,}25+0{,}2=0{,}6,\ \ j=2;$$
$$p(y_3)= 0+0{,}1+0{,}15=0{,}25,\ \ j=3.$$

Перевіряємо умову нормування:

$$p(y_1)+ p(y_2)+ p(y_3)=0{,}15+0{,}6+0{,}25=1.$$

Матрицю умовних ймовірностей знаходимо, скориставшись формулою множення ймовірностей $p(x_i, y_j)=p(y_j)\cdot p(x_i/y_j)$.

Звідси випливає, що $p(x_i / y_j) = \dfrac{p(x_i, y_j)}{p(y_j)}$.

Отже, матриця умовних ймовірностей $p(x_i/y_j)$ знаходиться так:

$$p(x_i / y_j) = \begin{bmatrix} \frac{0{,}15}{0{,}15} & \frac{0{,}15}{0{,}6} & \frac{0}{0{,}25} \\ \frac{0}{0{,}15} & \frac{0{,}25}{0{,}6} & \frac{0{,}1}{0{,}25} \\ \frac{0}{0{,}15} & \frac{0{,}2}{0{,}6} & \frac{0{,}15}{0{,}25} \end{bmatrix} = \begin{bmatrix} 1 & 0{,}25 & 0 \\ 0 & 0{,}4167 & 0{,}4 \\ 0 & 0{,}3333 & 0{,}6 \end{bmatrix}.$$
$$y_1 \quad y_2 \quad y_3$$

Для матриці умовних ймовірностей $p(x_i/y_j)$ повинна виконуватися умова нормування

$$\sum_i p(x_i / y_j) = 1.$$

Перевіряємо цю умову:

$$\sum_i p(x_i / y_1) = 1 + 0 + 0 = 1,$$
$$\sum_i p(x_i / y_2) = 0{,}25 + 0{,}42 + 0{,}33 = 1,$$
$$\sum_i p(x_i / y_3) = 0 + 0{,}4 + 0{,}6 = 1.$$

Скориставшись матрицею умовний ймовірностей $p(x_i/y_j)$, обчислимо часткові умовні ентропії X стосовно Y:

$$H(X / y_1) = -\sum_i p(x_i / y_1)\log_2 p(x_i / y_1) = -\log_2 1 = 0\ \ (\textit{біт/сим});$$

$$H(X / y_2) = -\sum_i p(x_i / y_2)\log_2 p(x_i / y_2) = -\left(0{,}25\log_2 0{,}25 + \right.$$
$$\left. + 0{,}42\log_2 0{,}42 + 0{,}33\log_2 0{,}33\right) \approx 1{,}555\ (\textit{біт/сим});$$

$$H(X / y_3) = -\sum_i p(x_i / y_3)\log_2 p(x_i / y_3) = -\left(0 + 0{,}4\log_2 0{,}4 + \right. \qquad \left. + 0{,}6\log_2 0{,}6\right) \approx 0{,}971\ (\textit{біт/сим}).$$

Виходячи з безумовного закону розподілу д. в. в. Y та знайдених часткових умовних ентропій $H(X/y_j)$, відшукуємо їх математичне сподівання – загальну умовну ентропію

$$\boldsymbol{H}(\boldsymbol{X}/\boldsymbol{Y}) = \sum_j \boldsymbol{p}(\boldsymbol{y}_j) \cdot \boldsymbol{H}(\boldsymbol{X}/\boldsymbol{y}_j) = 0{,}15 \cdot 0 + 0{,}6 \cdot 1{,}555 + 0{,}25 \cdot 0{,}971 \approx$$

$$\approx 1{,}176 \text{ (}\textit{біт/сим}\text{)}.$$

Отже, ***інформаційні втрати*** в каналі зв'язку $\boldsymbol{H}(\boldsymbol{X}/\boldsymbol{Y}) \approx 1{,}18$ (*біт*/*сим*).

Пропускна здатність каналу із шумом обчислюється за формулою

$$\boldsymbol{C} = \frac{1}{\boldsymbol{\tau}}\left[\log_2 \boldsymbol{k} - \boldsymbol{H}(\boldsymbol{X}/\boldsymbol{Y})\right],$$

де через $\boldsymbol{k}$ позначено об'єм алфавіту джерела; $\boldsymbol{\tau}$ - час вибору повідомлення джерелом.

Отже, отримаємо $\boldsymbol{C} = \frac{1}{10^{-3}}\left[\log_2 3 - 1{,}176\right] \approx 0{,}409 \cdot 10^3 = 409$ (*бод*).

Кількість переданої по каналу інформації, що припадає на одне повідомлення джерела, знаходиться, виходячи із середньої кількості інформації, що виробляється джерелом – його ентропії і інформаційних втрат в каналі:

$$\boldsymbol{I}(\boldsymbol{X}, \boldsymbol{Y}) = \boldsymbol{HX} - \boldsymbol{H}(\boldsymbol{X}/\boldsymbol{Y}) = 1{,}581 - 1{,}176 \approx 0{,}406 \text{ (}\textit{біт/сим}\text{)}.$$

Швидкість передачі інформації знаходиться так:

$$\boldsymbol{v} = \frac{\boldsymbol{I}(\boldsymbol{X}, \boldsymbol{Y})}{\boldsymbol{\tau}} = 0{,}406 \cdot 10^3 = 406 \text{ (}\textit{бод}\text{)}.$$

Відповідь: $\boldsymbol{H}(\boldsymbol{X}/\boldsymbol{Y}) \approx 1{,}18$ (*біт*/*сим*); $\boldsymbol{C} \approx 409$ (*бод*); $\boldsymbol{v} = 406$ (*бод*).

4. Математичні моделі джерел неперервних повідомлень та їх статистичні характеристики

У телекомунікаційних системах неперервне повідомлення $a(t)$ перетворюється пропорційно (без втрат інформації) у первинний сигнал $b(t) = ka(t)$. Виявилось зручним замість аналізу інформаційних характеристик джерела неперервних повідомлень аналізувати

інформаційні характеристики первинного сигналу $b(t)$. Тому в подальшому мова йде лише про первинний неперервний сигнал $b(t)$.

Сигнал $b(t)$ – це реалізації стаціонарного ергодичного випадкового процесу $B(t)$ з його ймовірнісними характеристиками:

1) одновимірними функцією розподілу $F(b)$ та густиною ймовірності $p(b)$;
2) середнім значенням B та дисперсією $D\{B\}$;
3) функцією кореляції $KB(\tau)$ та спектральною густиною потужності $GB(\tau)$.

Всі неперервні повідомлення (еквівалентні їм дійсні первинні сигнали) мають ***спектри***, зосереджені в обмеженій смузі частот 0...Fmax.

Таблиця 6.1 - Розрахункові формули інформаційних характеристик неперервного сигналу $b(t)$

Найменування характеристики	**Розрахункова формула**
Кількість інформації у відліку, дв.од. (біт)	Прямує до нескінченості
Диференціальна ентропія джерела при незалежних відліках, дв.од./відлік (біт/відлік)	$h(B) = -\int_{-\infty}^{\infty} p(b)\log_2(p(b))db$
Взаємна диференціальна ентропія джерел $b(t)$ і $\hat{b}(t)$ при незалежних відліках, дв.од./відлік (біт/відлік)	$h_{вз}(B/\hat{B}) = h(B) - h(B/\hat{B}) =$ $= h(\hat{B}) - h(\hat{B}/B)$
Максимальна диференціальна ентропія при обмеженій середній потужності сигналу $b(t)$, дв.од./відлік (біт/відлік)	$h_{max}(B) = \log_2\sqrt{2\pi eD\{B\}}$
Максимальна диференціальна ентропія при обмеженому інтервалі значень сигналу $b(t)$, дв.од./відлік (біт/відлік)	$h_{max}(B) = \log_2\sqrt{12D\{B\}}$
Відносна ентропія неперервного сигналу $b(t)$, який відтворено з похибкою $\varepsilon(t)$, дв.од./відлік (біт/відлік)	$H_{від}(B) = h(B) - h(\text{E})$
Епсилон-ентропія неперервного сигналу $b(t)$, дв.од./відлік (біт/відлік)	$H_\varepsilon(B) = \min[h(B) - h(\text{E})] =$ $= h(B) - \max h(\text{E})$
Надмірність (надлишковість) джерела	$K_{над} = 1 - H_\varepsilon(B)/\max H_\varepsilon(B)$
Епсилон-продуктивність джерела, дв.од./с (біт/с)	$R_{\varepsilon\,дж} = 2F_{max}H_\varepsilon(B)$
Пояснення: $D\{B\} = \sigma_b^2$ – дисперсія сигналу $b(t)$; $p(b)$ – одновимірна густина ймовірності відліку сигналу $b(t)$; F_{max} – максимальна частота спектра сигналу $b(t)$	

1. Інформаційні характеристики джерел неперервних повідомлень

Результат не є несподіваним – кількість інформації в неперервному по-відомленні і відповідному йому первинному сигналі $b(t)$ прямує до нескінченності. А це тому, що

неперервне повідомлення має нескінченну множину реалізацій, імовірність появи будь-якої з них прямує до нуля.

Інтуїтивно зрозуміло, що різні сигнали містять різну кількість інформації.

Крім того, сигнал більшої тривалості надає одержувачу більшу кількість інфо-рмації. Тому для числової оцінки середньої кількості інформації неперервного джерела (за аналогією з дискретним джерелом) було введено поняття "***ентропія неперервного джерела***".

Диференціальна ентропія джерела неперервного сигналу $b(t)$ є аналогом ентропії джерела дискретного повідомлення, формально обчислюється як ма-тематичне сподівання кількості інформації у відліку за густиною ймовірності, характеризує ступінь невизначеності джерела. Звертаємо увагу на те, що диференціальна ентропія:

- обчислюється на відлік, а відліки сигналу можуть бути незалежними або залежними;
- залежить від дисперсії сигналу $b(t)$ та її розмірності;
- не показує середньої кількості інформації у відліку, але надає можливість порівнювати кількість інформації різних джерел.

Таблиця 6.2 - Розрахункові формули для диференціальної ентропії (на відлік) сигналу $b(t)$ з дисперсією $D\{B\}$ при незалежних відліках, дв.од./відлік (біт/відлік)

Розподіл імовірностей сигналу $b(t)$	**Диференціальна ентропія $h(B)$**
Гауссів: $p(b)=\frac{1}{\sqrt{2\pi D\{B\}}}\exp\left(-\frac{b^2}{2D\{B\}}\right)$	$\log_2\sqrt{2\pi eD\{B\}}$
Односторонній експоненціальний: $p(b)=\begin{cases}\frac{1}{\sqrt{D\{B\}}}\exp\left(-\frac{b}{\sqrt{D\{B\}}}\right), & b\geq 0,\\ 0, & b<0\end{cases}$	$\log_2\sqrt{e^2D\{B\}}$
Двосторонній експоненціальний (Лапласа): $p(b)=\frac{1}{\sqrt{2D\{B\}}}\exp\left(-\frac{\sqrt{2}\lvert b\rvert}{\sqrt{D\{B\}}}\right)$	$\log_2\sqrt{2e^2D\{B\}}$
Рівномірний: $p(b)=\begin{cases}1/(2b_{\max}), & \lvert b\rvert\leq b_{\max}\\ 0, & \lvert b\rvert>b_{\max}\end{cases}$	$\log_2\sqrt{12D\{B\}}$

Ентропію джерела неперервного сигналу b(t), яка характеризує кількість інформації у відліку, можна обчислити за різними співвідношеннями. Для цього треба звернутись до того очевидного факту, що сигнал $b(t)$ завжди можна подати наближеним до нього сигналом $\hat{b}(t)$ з деякою похибкою $\varepsilon(t)=\hat{b}(t)-b(t)$. Допустимий середній квадрат похибки $\overline{\varepsilon^2(t)}$ можна задати.

Відносна ентропія (синонім – ентропія Кульбака-Лейблера) [5] неперервного сигналу обчислюється як взаємна ентропія між сигнал $b(t)$ та його наближеним поданням $\hat{b}(t)$ (формула (4.6)). При цьому ніяких вимог до функції розподілу похибки $\varepsilon(t)$ не пред'являється, але очевидно, що чим більший середній квадрат похибки $\overline{\varepsilon^2(t)}$, тим менше значення відносної ентропії.

Академік А.М. Колмогоров увів поняття ***епсилон-ентропії***, використавши поняття еквівалентність сигналу $b(t)$ і його наближеного подання $\hat{b}(t)$. Сигнали $b(t)$ і $\hat{b}(t)$ називаються ***еквівалентними***, якщо середній квадрат похибки $\overline{\varepsilon^2(t)}$ не перевищує задане число ε_0^2.

Епсилон-ентропією $H_\varepsilon(B)$ називається мінімальна середня кількість інформації в одному незалежному відліку сигналу $\hat{b}(t)$ відносно сигналу $b(t)$, коли вони еквівалентні при заданому значенні похибки ε_0^2. При цьому розподіл імовірності похибки $\varepsilon(t)$ має бути таким, що забезпечує мінімальну середню кількість інформації у відліку (за обмеженої дисперсії похибки це гауссів розподіл імовірності).

Таблиця 6.3 - Розрахункові формули епсилон-ентропії на відлік неперервного сигналу $b(t)$ при незалежних відліках

Розподіл імовірностей сигналу $b(t)$	**Епсилон-ентропія $H_\varepsilon(B)$, дв.од./відлік (біт/відлік)**
Гауссів	$0{,}5\cdot\log_2 c_{с/п}$,
Односторонній експоненціальний	$0{,}5\cdot\log_2[(e/2p)\rho_{с/п}]$
Двосторонній експоненціальний (Лапласа)	$0{,}5\cdot\log_2[(2e/p)c_{с/п}]$
Рівномірний	$0{,}5\cdot\log_2((6/e\pi)\rho_{с/п})$
Пояснення: $c_{с/п}$ – відношення дисперсій сигналу $b(t)$ і похибки $\varepsilon(t)$	

Продуктивність (швидкість видачі інформації) джерела неперервного сигналу можна обчислити, знаючи ентропію.

Продуктивність, обчислена за відносною ентропією, дістала назву – ***функція швидкість-спотворення***

Тема 4. Кодування повідомлень

План

1. **Код Шеннона-Фано**
2. **Код Хаффмена**

Оптимальним кодуванням називають процедуру перетворення символів первинного алфавіту *A* на кодові комбінації вторинного алфавіту *B*, при якій середня довжина повідомлення у вторинному алфавіті мінімальна.

Знайти код, який був би оптимальним з усіх точок зору, практично неможливо. Тому код може бути оптимальним тільки за певних умов (з точки зору швидкості передачі інформації, здатності виправляти помилки тощо). Існує кілька методик побудови оптимальних кодів з точки зору швидкості передачі інформації. До цих кодів належать оптимальні нерівномірні коди (ОНК), які передають повідомлення комбінаціями мінімальної середньої довжини.

1. Код Шеннона-Фано

Алгоритм побудови ОНК даного методу має такі кроки.

1. Множину з N повідомлень для кодування розміщують у порядку спадання ймовірностей.

2. Упорядковані за ймовірностями повідомлення розбивають по можливості на q рівноймовірних груп.

3. Кожній із груп завжди в одній і тій самій послідовності приписують символи алфавіту q (всім повідомленням першої групи – першу якісну ознаку цього алфавіту, всім повідомленням другої групи – другу якісну його ознаку тощо).

4. Створені групи розбивають по можливості на рівноймовірні підгрупи, кількість яких дорівнює або менша ніж q (якщо пісня розбивання у групі залишається одне повідомлення, то подальший поділ стає неможливим).

5. Кожній із утворених підгруп присвоюють якісні ознаки з алфавіту q, використовуючи крок алгоритму 3.

6. Розбивання та присвоєння ознак алфавіту q повторюють до того часу, поки після чергового поділу в утворених підгрупах залишиться не більш ніж одне повідомлення.

Приклад 6.1.1. Закодувати 16 повідомлень за допомогою четвіркового ОНК алфавітом $q = 4$, якщо повідомлення на виході джерела з'являються з ймовірностями $P(x_1) = 0{,}22$; $P(x_2) = P(x_3) = 0{,}1$; $P(x_4) = 0{,}08$; $P(x_5) = P(x_6) = 0{,}07$; $P(x_7) = P(x_8) = 0{,}06$; $P(x_9) = P(x_{10}) = 0{,}05$; $P(x_{11}) = 0{,}04$; $P(x_{12}) = 0{,}03$; $P(x_{13}) = P(x_{14}) = P(x_{15}) = 0{,}02$; $P(x_{16}) = 0{,}01$.

Розв'язання. Використовуючи дані прикладу, кроки 1 і 2 алгоритму, будуємо таблицю для знаходження кодових комбінацій ОНК, табл.6.1.1.

Таблиця 8.1

Номер повідомлення	Ймовірність повідомлення	Поділ групи				Кодові комбінації ОНК
		1	2	3	4	
1	0,22	→				0
2	0,1	→	→			10
3	0.1	→	→			11
4	0,08	→	→			12
5	0,07	→		→		20
6	0,07	→		→		21
7	0,06	→		→		22
8	0,06	→		→		23
9	0,05	→			→	30
10	0,05	→			→	31
11	0,04	→			→	32
12	0,03	→			→	33
13	0.02	→			→	330
14	0,02	→			→	331
15	0,02	→			→	332
16	0,01	→			→	333

Згідно з другим алгоритмом визначаємо квант поділу повідомлень за ймовірностями, який буде дорівнювати $1/q = 1/4 = 0.25$, і розбиваємо всі повідомлення по можливості на чотири рівномірні групи:

$\sum_{i=1}^{1} P(x_i) = 0{,}22$; $\sum_{i=2}^{4} P(x_i) = 0{,}28$; $\sum_{i=5}^{8} P(x_i) = 0{,}26$; $\sum_{i=9}^{16} P(x_i) = 0{,}24$.

Користуючись третім кроком алгоритму, завжди присвоюємо кожній із груп в одній і тій самій послідовності символи алфавіту q (першому повідомленню першої групи першу якісну ознаку 0 ; всім повідомленням другої групи – другу якісну ознаку 1 ; всім повідомленням третьої групи – третю якісну ознаку 2 ; всім повідомленням четвертої групи – четверту якісну ознаку 3. Дані про якісні ознаки заносимо в табл.6.1.1 як першу цифру стовпчика комбінацій ОНК.

Використовуючи четвертий і п’ятий кроки алгоритму, присвоюємо кожній із утворених підгруп завжди в одній і тій самій послідовності якісні ознаки з алфавіту q згідно з третім кроком алгоритму (0,1,2 – відповідні повідомлення першої і другої груп; 0,1,2,3 – відповідні повідомлення третьої і четвертої груп). Дані якісні ознаки заносимо в табл.6.1.1 як першу, другу і третю цифри стовпчика комбінації ОНК.

Користуючись кроками 4, 6 алгоритму, останнім чотирьом повідомленням присвоюємо якісні ознаки 0, 1, 2, 3 з алфавіту q згідно з третім кроком алгоритму. Дані якісні ознаки заносимо в таблицю 6.1.1 як третю цифру стовпчика комбінації ОНК.

Для перевірки оптимальності коду щодо довжини кодових комбінацій необхідно визначити середню довжину $n_{сер}$ кодової комбінації ОНК. У разі оптимальності ця довжина не повинна перевищувати довжини рівномірного четвіркового коду, яким можна закодувати 16 повідомлень, тобто $q_n = 4^2 = 16$ $(n = 2)$.

$n_{сер} = \sum_{i=1}^{16} P(x_i) \cdot n_i = 0{,}22 \cdot 1 + (0{,}1 + 0{,}1 + 0{,}08 + 0{,}07 + 0{,}07 + 0{,}06 + 0{,}06 + 0{,}05 + +0{,}05 + 0{,}04$
$+ 0{,}03)\ 2 + (0{,}02 + 0{,}02 + 0{,}02 + 0{,}02 + 0{,}01)\ 3 = 0{,}22 + 1{,}42 + 0{,}21 = 1{,}85 < 2.$ Таким чином, отриманий ОНК є дійсно оптимальний, оскільки $n_{сер} < n$.

2. Код Хаффмена

Дана методика, як і методика Шеннона - Фано, передбачає побудову ОНК у кодовому алфавіті з кількістю якісних значень q. Згідно з цією методикою алгоритм побудови ОНК має такі кроки.

1. Множину з N повідомлень, що кодують , розташовують у порядку спадання ймовірностей.

2. Останні N_o повідомлень $2 \le N_o \le q$ об'єднують у нове повідомлення з імовірністю, що дорівнює сумі ймовірності об'єднуваних повідомлень.

3. Утворену множину $(N - N_o + 1)$ повідомлень розташовують у порядку спадання ймовірностей.

4. Об'єднують останні q повідомлень і впорядковують множину повідомлень у порядку спадання ймовірностей. Так діють до того часу, поки ймовірність чергового об'єднаного повідомлення не дорівнюватиме одиниці.

5. Будують кодове дерево, починаючи з кореня, і гілкам цього дерева присвоюють якісні ознаки кодового алфавіту q. Кодові комбінації ОНК – це послідовність якісних ознак, які зустрічаються на шляху від кореня до вершини кодового дерева.

Приклад 6.2.1. Закодувати 16 повідомлень комбінаціями четвіркового ОНК з ймовірностями появи посилань коду: $P(x_1) = 0{,}22$; $P(x_2) = P(x_3) = 0{,}1$; $P(x_4) = 0{,}08$; $P(x_5) = P(x_6) = 0{,}07$; $P(x_7) = P(x_8) = 0{,}06$; $P(x_9) = P(x_{10}) = 0{,}05$; $P(x_{11}) = 0{,}04$; $P(x_{12}) = 0{,}03$; $P(x_{13}) = P(x_{14}) = P(x_{15}) = 0{,}02$; $P(x_{16}) = 0{,}01$;

Розв'язання. Використовуючи дані прикладу і крок 1 алгоритму, будуємо таблицю для знаходження кодових комбінацій ОНК, табл.6.2.1.

Таблиця 8.2

Номер повідом-лення	Ймовірність повідомлення $P(Xi)$	Ймовірності повідомлень при об'єднаннях					Кодові комбінації ОНК
		першому	другому	третьому	четвер-тому	п'ятому	
1	0,22	0,22	0,22	0,26	0,35	1,0	2
2	0,1	0,1	0,17	0,22	0,26		00
3	0,1	0,1	0,1	0,17	0,22		01
4	0,08	0,08	0,1	0,1	0,17		02
5	0,07	0,07	0,08	0,1			03
6	0,07	0,07	0,07	0,08			10
7	0,06	0,07	0,07	0,07			12
8	0,06	0,06	0,07				13
9	0,05	0,06	0,06				30
10	0,05	0,05	0,06				31
11	0,04	0,05					32
12	0,03	0,04					33
13	0,02	0,03					110
14	0,02						111
15	0,02						112
16	0,01						113

Виконуючи крок 2 алгоритму, об'єднуємо останні чотири повідомлення (оскільки $q = 4$) й утворюємо нове умовне повідомлення. Користуючись третім кроком алгоритму, утворену множину розташовуємо в порядку спадання ймовірностей.

Згідно з четвертим кроком алгоритму знову об'єднуємо останні чотири повідомлення та впорядковуємо множину повідомлень у порядку спадання ймовірностей. Цю процедуру повторюємо ще три рази, поки при останньому об'єднанні сумарна ймовірність не досягне значення одиниці.

Використовуючи п'ятий крок алгоритму, будуємо кодове дерево, рис. 8.1.

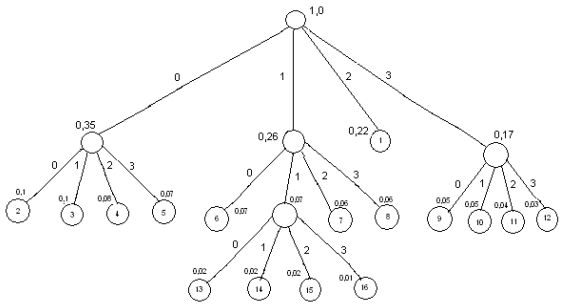


Рисунок 8.1

Гілкам кадрового дерева присвоюємо якісні ознаки кодового алфавіту від 0 до 3. Кодові комбінації ОНК для кожного повідомлення заносимо до табл.6.2.1. Вони визначаються послідовністю якісних ознак, які зустрічаються на шляху від кореня до певної вершини кодового дерева.

Перевірку на оптимальність коду щодо довжини кодових комбінацій робимо аналогічно, як це наведено у прикладі 6.1.1.

$$n_{сер} = \sum_{i=1}^{16} P(x_i) \cdot n_i = 0{,}22 \cdot 1 + (0{,}1 + 0{,}1 + 0{,}08 + 0{,}07 +$$

$$+ 0{,}07 + 0{,}06 + 0{,}06 + 0{,}05 + 0{,}05 + 0{,}04 + 0{,}031) \cdot 2 + (0{,}02 +$$

$$+ 0{,}02 + 0{,}02 + 0{,}01) \cdot 3 = 0{,}22 + 1{,}42 + 0{,}21 = 1{,}85 < 2.$$

Таким чином, отриманий ОНК є дійсно оптимальним, оскільки $n_{сер} < n$.

До недоліків алгоритму Хаффмена побудови ОНК слід віднести громіздкість (особливо зі збільшенням кількості повідомлень N та алфавіту q коду), що пояснюється необхідністю побудови кодового дерева.

Тема 5. Ефективне кодування

План

1. **Теоретичні границі стиснення інформації**
2. **Метод блокування повідомлення**

1. Теоретичні границі стиснення інформації

Стиснення даних не може бути більше деякої теоретичної границі. Сформульована раніше теорема *Шеннона* про кодування каналу без шуму встановлює ***верхню границю стиснення інформації як ентропію джерела H(X).***

Позначимо через ***L(X)*** функцію, що повертає довжину коду повідомлень

$$L(X)=len(code(X)),$$

де ***code(X)*** кожному значенню ***X*** ставить у відповідність деякий бітовий код; ***len( )*** - повертає довжину цього коду.

Оскільки ***L(X)*** - функція від д. в. в. ***X***, тобто також є д. в. в., то її середнє значення обчислюється як математичне сподівання:

$$\overline{L(X)} = \sum_{i=1}^{k} p(x_i) L(X_i). \qquad (9.1)$$

Наслідком теореми Шеннона про кодування джерела у відсутності шуму є те, що ***середня кількість бітів коду, що припадає на одне значення д. в. в., не може бути менше її ентропії***, тобто

$$\overline{L(X)} \geq HX \qquad (9.2)$$

для будь-якої д. в. в. ***X*** і будь-якого її коду.

Нехай $\vec{X} = (x_1, x_2, \ldots, x_n)$ - вектор даних завдовжки n; $\vec{F} = (F_1, F_2, \ldots, F_k)$ - вектор частот символів у $\vec{X}$.

Тоді середня кількість бітів коду на одиницю повідомлення $\vec{X}$ обчислюється так:

$$\overline{L(X)} = \frac{L(\vec{X})}{n} = \frac{1}{n}\sum_{i=1}^{k} F_i L_i, \qquad (9.3)$$

де $L(\vec{X})$ - довжина коду повідомлення $\vec{X}$: $L(\vec{X}) = len(code(\vec{X}))$; $(L_1, L_2, \ldots, L_k)$ - вектор *Крафта* для $\vec{X}$.

Ентропія повідомлення $\vec{X}$ обчислюється так:

$$HX \cong \frac{1}{n}\sum_{i=1}^{k} F_i \log_2 \frac{n}{F_i}. \qquad (9.4)$$

Розглянемо функцію ***y=ln(x)***, яка є опуклою вниз, тобто її графік лежить не вище своєї дотичної ***y=x-1***. Тоді можна записати таку нерівність:

$$ln(x) \leq x\text{-}1,\ x>0. \qquad (9.5)$$

Підставимо в (2.7) $x = \frac{n \cdot 2^{-L_i}}{F_i}$ і помножимо обидві частини цієї нерівності на $\frac{F_i}{n}$:

$$\frac{F_i}{n}\cdot\ln\frac{2^{-L_i}}{\frac{F_i}{n}}\le 2^{-L_i}-\frac{F_i}{n}. \qquad (9.6)$$

Запишемо суми за i обох частин нерівності (2.8), і з урахуванням того, що для оптимального кодування *Хаффмена* нерівність *Крафта* (2.1) переходить в строгу рівність, дістанемо в правій частині (2.8) нуль, отже,

$$\sum_i \frac{F_i}{n}\ln\left(2^{-L_i}\right)-\sum_i \frac{F_i}{n}\ln\left(\frac{F_i}{n}\right)\le 0.$$

Перейдемо від натурального логарифма до двійкового і з урахуванням виразу (9.6) дістанемо

$$-\overline{L(X)}+H(X)\le 0,$$

тобто приходимо до виразу (2.4), що визначає ***верхню границю стиснення даних***.

Припустимо, що у векторі *Крафта* $(L_1^*, L_2^*, \dots, L_k^*)$ довжини кодових слів пов'язані з частотами символів у повідомленні так:

$$L_l^* \cong -\log_2\left(\frac{F_i}{n}\right).$$

При виконанні цієї умови нерівність *Крафта* (2.1) обертається у строгу рівність, і код буде *компактним*, тобто матиме найменшу середню довжину. Тоді, оскільки для оптимальних кодів *Шеннона-Фано* і *Хаффмена* довжина кожної кодової комбінації округлюється до більшої цілої кількості бітів, маємо

$$\overline{L(X)}=\frac{1}{n}\sum_i F_i L_i^* \le \frac{1}{n}\sum_i F_i\left(-\log_2\left(\frac{F_i}{n}\right)+1\right),$$

звідси

$$\overline{L(X)}\le \frac{1}{n}\sum_i F_i\log_2\left(\frac{n}{F_i}\right)+\frac{1}{n}\sum_i F_i = HX+1.$$

Таким чином, ***границі стиснення інформації*** при *оптимальному статистичному кодуванні* визначаються так:

$$HX\le \overline{L(X)}\le HX+1. \qquad (9.7)$$

2. Метод блокування повідомлення

На відміну від раніше розглянутих методів кодування ***блокові коди*** належать до так званих ***кодів з пам'яттю***, оскільки при кодуванні поточного символу враховуються значення одного або декількох попередніх символів.

Блоковий код *розділяє вектор даних на блоки певної довжини, і потім кожний блок замінює кодовим словом з префіксної множини кодових слів*. Отриману послідовність кодових слів об'єднують в остаточну двійкову послідовність на виході кодера.

Блоковий код називається ***блоковим кодом k-го порядку***, якщо всі його блоки мають довжину k символів.

Метод блокування повідомлень полягає в такому.

За заданим $\varepsilon>0$ можемо знайти таке k, що якщо розбити повідомлення на блоки завдовжки k (всього буде n/k блоків), то, використовуючи оптимальне статистичне кодування таких блоків, що розглядаються як одиниці повідомлень, можна досягти середньої довжини коду більше ентропії менш ніж на ε.

Припустимо, що X_1, X_2, …, X_n – незалежні д. в. в., що мають однаковий розподіл ймовірностей. Тоді ентропія n-вимірної д. в. в. $\vec{X} = (X_1, X_2, \dots, X_n)$

$$H\vec{X} = nHX_1 .$$

Нехай $\vec{Y} = (\vec{Y}_1, \vec{Y}_2, \dots, \vec{Y}_{n/k})$ - повідомлення джерела, де $\vec{Y}_1 = (X_1, X_2, \dots, X_k)$, $\vec{Y}_2 = (X_{k+1}, X_{k+2}, \dots, X_{2k})$, … , $\vec{Y}_i = (X_{k(i-1)+1}, X_{k(i-1)+2}, \dots, X_{ki})$ - блоки повідомлення. Тоді

$$H\vec{Y}_1 = kHX_1 . \qquad (9.9)$$

При *оптимальному кодуванні* k-послідовностей векторної д. в. в. $\vec{Y}$, що розглядаються як одиниці повідомлення, справедлива нерівність

$$\overline{L(\vec{Y}_1)} \le H\vec{Y}_1 + 1 . \qquad (9.10)$$

Середня кількість бітів на одиницю повідомлення X

$$\overline{L(X)} = \frac{\overline{L(\vec{Y}_i)}}{k} . \qquad (9.11)$$

Тоді з урахуванням (2.10) і (2.12) нерівність (2.11) можна записати так:

$$k\overline{L(X)} \le kHX + 1 . \qquad (9.12)$$

Розділивши обидві частини (2.13) на k, отримуємо

$$\overline{L(X)} \le HX + \frac{1}{k} , \qquad (9.13)$$

тобто достатньо вибрати $k = 1/\varepsilon$.

Приклад 1 Стиснемо вектор даних X=(0, 1, 1, 0, 1, 0, 0, 1, 1, 1, 1, 0, 0, 1, 0, 1), використовуючи блоковий код 2-го порядку.

Розіб'ємо вектор на блоки довжини 2: (01, 10, 10, 01, 11, 10, 01, 01). Розглядатимемо ці блоки як елементи нового алфавіту {01, 10, 11}. Визначимо вектор частот появи блокових елементів в заданій послідовності. Одержуємо (4, 3, 1), тобто найбільш часто трапляється блок 01, потім 10 і найменше - 11; всього 8 блоків. Код *Хаффмена* для блоків символів представимо у вигляді кодового дерева (*рис. 2*) і відповідної таблиці кодів (*табл. 2.5*).

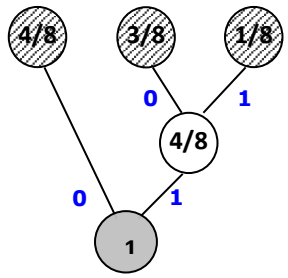


Рисунок 2. 7

Таблиця 2.5

Блок повідомлення	Кодове слово
01	0
10	10
11	11

Замінюючи кожний блок повідомлення відповідним кодовим словом з таблиці кодів, дістанемо вихідну кодову послідовність ***Code***(**01**, **10**, **10**, **01**, **11**, **10**, **01**, **01**)= **010100111000**.

Обчислимо ентропію, використовуючи статистичний розподіл ймовірностей символів повідомлення:

$$\boldsymbol{HX} = \frac{7}{16}\log_2\frac{16}{7} + \frac{9}{16}\log_2\frac{16}{9} \approx 0{,}986 \ (\textit{біт/сим}).$$

Швидкість стиснення даних (середня довжина коду)

$$\overline{\boldsymbol{L(X)}} = \frac{\boldsymbol{k}}{\boldsymbol{n}} = \frac{12}{16} = 0{,}75 \ (\textit{біт/сим}).$$

Отриманий результат виявляється меншим за ентропію, що, здавалося, суперечить теоремі *Шеннона*. Проте це не так. Легко бачити з вектора початкових даних, що символ **0** частіше слідує **1**, тобто умовна імовірність ***p*(1/0)** більше безумовної ***p*(1)**. Отже, ентропію цього джерела необхідно обчислювати як ентропію статистично взаємозалежних елементів повідомлення, а вона менша за безумовну.

Приклад 2 Нехай д. в. в. $X_1, X_2, \ldots, X_n$ незалежні, однаково розподілені і можуть набувати тільки два значення **0** та **1** з такою ймовірністю: $P(X_i=0)=3/4$ та $P(X_i=1)=1/4$, $i=1\ldots n$.

Тоді ентропія одиниці повідомлення

$$\boldsymbol{HX} = \frac{3}{4}\log_2\frac{4}{3} + \frac{1}{4}\log_2 4 = 2 - \frac{3}{4}\log_2 3 \approx 0{,}811 \ (\textit{біт/сим}).$$

Мінімальне префіксне кодування - це коди **0** або **1** завдовжки 1 біт. Отже, середня кількість бітів на одиницю повідомлення $\overline{\boldsymbol{L(X)}} = 1$ (*біт/сим*).

Розіб'ємо повідомлення на блоки довжиною $n=2$. Закон розподілу ймовірностей і відповідне кодування двовимірної д. в. в. $\vec{X} = (X_1, X_2)$ наведені у *табл. 2.6*.

Таблиця 2.6

$\vec{X}$	**00**	**01**	**10**	**11**
P	*9/16*	*3/16*	*3/16*	*1/16*
Префікс ний код	**0**	**10**	**110**	**111**
$L(\vec{X})$	1	2	3	3

У цьому випадку середня кількість бітів на одиницю повідомлення $\overline{\boldsymbol{L(X)}} = \overline{\boldsymbol{L(\vec{X})}}/\boldsymbol{n}$ =(1·9/16+2·3/16+3·3/16)/2=27/32≈
≈0,84375 (*біт/сим*), тобто менше, ніж для неблокового коду.

Для блоків довжини три середня кількість бітів на одиницю повідомлення $\overline{\boldsymbol{L(X)}} \approx 0{,}823$, для блоків довжини чотири – $\overline{\boldsymbol{L(X)}} \approx 0{,}818$ і т. д.

Метод блокування повідомлення

На відміну від раніше розглянутих методів кодування ***блокові коди*** належать до так званих ***кодів з пам'яттю***, оскільки при кодуванні поточного символу враховуються значення одного або декількох попередніх символів.

Блоковий код *розділяє вектор даних на блоки певної довжини, і потім кожний блок замінює кодовим словом з префіксної множини кодових слів*. Отриману послідовність кодових слів об'єднують в остаточну двійкову послідовність на виході кодера.

Блоковий код називається ***блоковим кодом k-го порядку***, якщо всі його блоки мають довжину ***k*** символів.

Метод блокування повідомлень полягає в такому.

*За заданим $\varepsilon>0$ можемо знайти таке **k**, що якщо розбити повідомлення на блоки завдовжки **k** (всього буде **n/k** блоків), то, використовуючи оптимальне статистичне кодування таких блоків, що розглядаються як одиниці повідомлень, можна досягти середньої довжини коду більше ентропії менш ніж на ε.*

Приклад_2. Згрупувати по два і по три повідомлення в групі. Побудувати код Хаффмана. Виконати порівняльну характеристику щодо ефективності коду, швидкості передачі та похибки коду. Значення ймовірностей наступні: $x_1 = 0.8$, $x_2 = 0.2$.

Розв'язання

Код Хаффмана для двох повідомлень в групі:

Знайдемо ентропію для заданих повідомлень:

$$H(x) = -\left(0.8 \cdot \log 0.8 + 0.2 \cdot \log 0.2\right) = 0.722 \ \frac{\text{біт}}{\text{повідомл.}}.$$

$x_i x_j$	$P(x_i x_j)$				Код	μ_i
$x_1 x_1$	0.64	0.64	0.64 (1)	1	1	1
$x_1 x_2$	0.16	0.20 (1)	0.36 (0)		00	2
$x_2 x_1$	0.16 (1)	0.16 (0)			011	3
$x_2 x_2$	0.04 (0)				010	3

Середня довжина кодового слова, яка припадає на одне повідомлення:

$$L = \frac{1}{2} \cdot \left(1 \cdot 0.64 + 2 \cdot 0.16 + 3 \cdot 0.6 + 3 \cdot 0.04\right) = 0.780.$$

Швидкість передачі повідомлення:

$$R_t = \frac{H(x)}{\tau} = \frac{0.722}{0.780 \cdot 10^{-6}} = 925641 \ \frac{\text{біт}}{\text{повідомл.}}.$$

Щоб знайти похибку коду, обчислимо ймовірність появи нулів і одиниць:

$$P(0) = \frac{2 \cdot 0.16 + 1 \cdot 0.16 + 2 \cdot 0.04}{2 \cdot 0.780} = 0.359, \ P(1) = 1 - P(0) = 0.641.$$

Тоді ентропія коду рівна:

$$H_к = -\left(0.359 \cdot \log 0.359 + 0.641 \cdot \log 0.641\right) = 0.942.$$

Похибка коду наступна:

$$R_к = 1 - H_к = 1 - 0.942 = 0.058.$$

Код Хаффмана для трьох повідомлень в групі:

$x_j x_i x_k$	$P(x_j x_i x_k)$							Код	μ_i
$x_1 x_1 x_1$	0.512	0.512	0.512	0.512	0.512	0.512	0.512 (1) 1	1	1
$x_1 x_1 x_2$	0.128	0.128	0.128	0.128	0.232	0.256 (1)	0.488 (0)	110	3
$x_1 x_2 x_1$	0.128	0.128	0.128	0.128	0.128 (1)	0.232 (0)		111	3
$x_2 x_1 x_1$	0.128	0.128	0.128	0.128 (1)	0.128 (0)			101	3
$x_1 x_2 x_2$	0.032	0.040	0.064 (1)	0.104 (0)				10010	5

$x_2x_1x_2$	0.032		0.032 1	0.040 0	10011	5
$x_2x_2x_1$	0.032	1	0.032 0		10001	5
$x_2x_2x_2$	0.008	0			10000	5

Середня довжина кодового слова, яка припадає на одне повідомлення:

$$L = \frac{1}{3} \cdot (1 \cdot 0.512 + 9 \cdot 0.128 + 15 \cdot 0.032 + 5 \cdot 0.008) = 0.728.$$

Швидкість передачі повідомлення:

$$R_t = \frac{H(x)}{\tau} = \frac{0.722}{0.728 \cdot 10^{-6}} = 991758 \frac{\text{біт}}{\text{повідомл.}}.$$

Щоб знайти похибку коду, обчислимо ймовірність появи нулів і одиниць:

$$P(0) = \frac{0.128 \cdot 2 + 0.032 \cdot 8 + 0.008 \cdot 4}{2 \cdot 0.728} = 0.374,\ P(1) = 1 - P(0) = 0.626.$$

Тоді ентропія коду рівна:

$$H_к = -(0.374 \cdot \log 0.374 + 0.626 \cdot \log 0.626) = 0.954.$$

Похибка коду наступна:

$$R_к = 1 - H_к = 1 - 0.954 = 0.046.$$

Результати кодування по два і по три повідомлення в групі методом Хаффмана наведені в наступній таблиці:

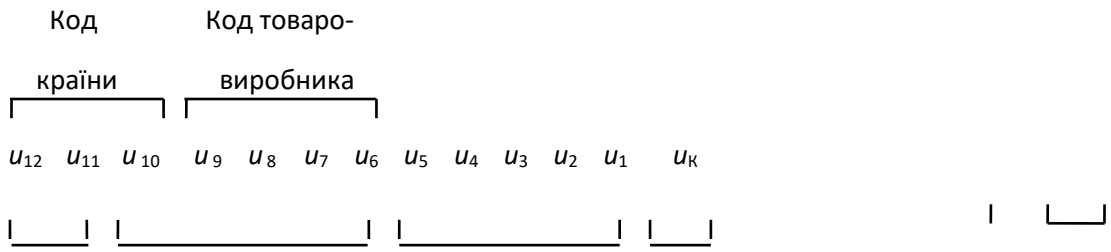
Обчислювані величини	Число повідомлень в групі		Граничні значення обчислюваних величин
	2	3	
L	0.780	0.728	$H(x)/\log 2 = 0.722$
R_t	925641	991758	$C = 1/\tau = 10^6$
$P(0)$	0.359	0.374	$P(0) = 0.5$
$P(1)$	0.641	0.626	$P(1) = 0.5$
$R_к$, %	5.8	4.6	$R_к = 0$

Тема 6. Штрихове та QR-кодування

Штрихові коди (ШК) широко використовуються в медицині, торгівлі, промисловості тощо. ШК – це послідовність штрихів і пробілів, що розташовані у напрямку уявленої прямої. Інформацію у ШК можуть нести як штрихи і пробіли різної ширини, так і штрихи різної висоти.

Існує багато ШК які в основному мають вузькоспеціальне призначення. Найбільш поширеними ШК є коди, що рекомендовані Міжнародною асоціацією EAN. Це коди EAN (European Article Numbering) та UPC (Uniform Product Code) .

Штрихові коди EAN призначені для кодування 10 цифр (0. . .9) і додаткових символів (СТАРТ, СТОП та розділові знаки). Код може мати довжину кодового слова 4, 5, 6, 7, 8, 10, 12, 13 і 14 знаків. Але існує два основні різновиди коду EAN: EAN-13 і EAN-8, де цифрою позначена довжина коду (кількість знаків у кодовому слові). Так наприклад, код EAN-13 має структуру, яка наведена на рис. 6.1.



Як видно з рисунка 6.1, код країни (див. табл. додатка *В*) може мати не два, а три знаки. У цьому разі код товаровиробника має не 5, а 4 знаки.

Контрольний символ визначається за таким алгоритмом:

1- ий крок: знаходять суму цифр, розташованих на непарних позиціях кодового слова (перегляд виконується справа наліво), і помножують одержаний результат на 3;

2 - ий крок: знаходять суму цифр, розташованих на парних позиціях кодового слова;

3 - ій крок: визначають добуток сум, знайдених при 1-му та 2-му кроках;

4 - ий крок: обчислюють контрольну цифру, яка дорівнює найменшому числу, що не перевищує 9, яке, якщо його додати до результату, одержаному на 3-му кроці, дає кратне 10 число.

Для кодування інформації у ШК EAN використовуються чотири набори знаків: *A*, *B*, *C* та *D* (табл.6.1) для кодування десяткових цифр, а також знаків СТАРТ, СТОП (*H*1, *H*2, *H*3), та розділових зна-ків (*H*4 та *H*5). Кожний знак містить у собі два штрихи і два пробіли. Довжина кожного знака для кодування цифр дорівнює 7 модулям (7-ми елементам зображення), а допоміжні знаки мають довжину 3, 5 і 6 модулів. Як знак СТАРТ використовуються набори *H*1 та *H*3, а знак СТОП – *H*1 та *H*2 у залежності від символів початку і кінця кодового слова (табл.6.2, де 0 – пробіл, 1 – штрих).

Таблиця 6.1

Сим-	Набір *A*	Набір *B*	Набір *C*	Набір *D*

воли	Двійковий еквівалент	Двійковий еквівалент	Двійковий еквівалент	Двійковий еквівалент
0	0001101	0100111	1110010	1011000
1	0011001	0110011	1100110	1001100
2	0010011	0011011	1101100	1100100
3	0111101	0100001	1000010	1011110
4	0100011	0011101	1011100	1100010
5	0110001	0111001	1001110	1000110
6	0101111	0000101	1010000	1111010
7	0111011	0010001	1000100	1101110
8	0110111	0001001	1001000	1110110
9	0001011	0010111	1110100	1101000
*H*1	101	Обмежувальні знаки СТАРТ і СТОП		
*H*2	010101			
*H*3	101010			
*H*4	01010	Розділові знаки		
*H*5	10101			

Примітка: При зображенні кодового слова у штриховому коді , згідно табл. 6.1, подають: 0 – одним, 00 – двома, 000 – трьома і 0000 – чотирма інтервалами, а 1 – тонким штрихом ( ) , 11 – штрихом ( ), 111 – штрихом ( ) і 1111 – штрихом ( ) .

Таблиця 6.2

Знак СТАРТ	Кодове слово	Знак СТОП
*H*1	0 0	*H*1
*H*3	1 1	*H*2
*H*1	0 1	*H*2
*H*3	1 0	*H*1

У штрихових кодах EAN довжиною 4, 5, 6, 7 знаків для кодування цифр використовується набір *A* (табл.6.1), а обмежувальних знаків – *H*1 (СТАРТ) та *H*2 (СТОП). У ШК довжиною 8, 10, 12 і 14 зна-ків кодові слова діляться на дві частини з однаковим числом знаків у кожній, які розділяються розділовим знаком *H*4. Для зображення знаків лівої частини кодового слова використовуються набори *A* і *B*, а правої – *C* і *D* (див. табл.6.1). Такі кодові слова мають обмежу-вальні знаки СТАРТ і СТОП типу *H*1.

У коді EAN-13 штрихове зображення складається з двох частин по шість знаків у кожній, які розділені знаком *H*4, і має зліва та справа обмежувальні знаки *H*1 (СТАРТ і СТОП). Першу зліва цифру (12-а цифра) товарного номера не кодують у вигляді штрихів і пробілів, а тільки пишуть зліва внизу. Ця цифра визначає спосіб кодування цифр, які розташовані у лівій

частині кодового слова між знаками *H*1 та *H*4 (табл.6.3). Літерами *A* і *B* у табл. 6.3 позначені набори з табл..6.1, якими кодують відповідні знаки лівої частини кодового слова. Цифри, які розташовані у правій частині кодового слова між знаками *H*4 і *H*1, кодують набором *C* (див. табл.6.1). Обмежувальні і розділовий знаки зображають більш довшими по висоті штрихами.

Таблиця 6.3

u_{12}	u_{11}	u_{10}	u_9	u_8	u_7	u_6
0	*A*	*A*	*A*	*A*	*A*	*A*
1	*A*	*A*	*B*	*A*	*B*	*B*
2	*A*	*A*	*B*	*B*	*A*	*B*
3	*A*	*A*	*B*	*B*	*B*	*A*
4	*A*	*B*	*A*	*A*	*B*	*B*
5	*A*	*B*	*B*	*A*	*A*	*B*
6	*A*	*B*	*B*	*B*	*A*	*A*
7	*A*	*B*	*A*	*B*	*A*	*A*
8	*A*	*B*	*A*	*B*	*B*	*A*
9	*A*	*B*	*B*	*A*	*B*	*A*

Деякі товари можуть мати короткий номер, що має 7 цифр. Після доповнення його контрольною цифрою, що виконується за наведеним вище алгоритмом для коду EAN-13, одержують 8 цифр, які кодують кодом EAN-8. Кодове слово EAN-8 складається зі знака СТАРТ – *H*1, чотирьох знаків набору *A*, розділового знака *H*4, трьох знаків набору *C*, знака контрольної цифри у наборі *C*, а також знака СТОП – *H*1. У коді EAN-8 перша цифра u_7 не визначає неявне кодування, а кодується як і наступні цифри u_6, u_5, u_4 набором *A* (див. табл.6.1).

Штрихові коди UPC використовуються у США та Канаді для ідентифікації товарів і також призначені для кодування 10 цифр та п’яти додаткових знаків. Коди UPC сумісні з кодами EAN з огляду на те, що використовують одну і ту ж таблицю наборів знаків (табл.6.1).

Існує декілька різновидів ШК UPC, з яких найбільше поширення набули коди UPC-A і UPC-E. Кодове слово ШК UPC-A має 12 цифр (12-а – контрольна), тобто на одну цифру менше ніж у EAN-13. Це викликано тим, що код країни (США і Канада) має тільки дві цифри. Відмінними ознаками коду UPC-A від коду EAN-13 є:

– старша цифра (u_{11}) товарного номера у штриховому зображенні кодується явно;

– штрихове зображення кодового слова у коді UPC-A містить знак СТАРТ (*H*1), 6 знаків набору *A*, розділовий знак (*H*4), 5 знаків набору *C*, знак контрольної суми набору *C* і знак СТОП (*H*1) (див. табл.6.1);

– у штриховому зображенні знаки цифр u_{11} і $u_К$ виконуються висотою, однаковою з висотою зображення знаків *H*1 і *H*4, причому значення цифр під цими знаками не позначають;

– зліва від штрихового зображення друкують цифру 0, що іденти-фікує код UPC-A.

Контрольна цифра визначається за тим же алгоритмом, що використовується і у коді EAN-13.

Товарний номер у коді UPC-E містить 6 цифр і також поділяється на дві частини по 3 цифри у кожній. Перша частина (ліва) кодується набором *A*, а друга (права) – набором *C* (у тому числі і контрольний знак) (див. табл.6.1).

При декодуванні ШК EAN та UPC перш за все визначається контрольний знак, який повинен співпадати з переданим. Крім того, сума всіх цифр кодового слова, що подається на приймальний пристрій, повинна бути кратною 10. У цьому разі помилки нема. Якщо ж сума буде не кратною 10, це вказує на наявність помилки. Помилка виявля-ється і при неправильному прийомі знаків, що не відповідають наборам, встановленим неявним кодуванням по старшому знаку в ШК EAN-13 цифр лівої половини кодового слова, тому що набори *A* і *B* (див. табл.6.1) не збігаються.

Задача 1.

Побудувати кодове слово у коді EAN-13, якщо країна товаровиробник – Україна, код товаровиробника –1229, код товару – 03458.

Розв'язання. 12 цифр кодового слова, яке треба закодувати кодом EAN-13, мають вигляд: 482122903458 (код країни – 482). Необхідно доповнити це кодове слово контрольною цифрою $u_К$, яку визначаємо згідно з алгоритмом для коду EAN:

а) визначаємо суму цифр, які розміщені на непарних місцях кодового слова (справа наліво): 8 + 4 + 0 + 2 + 1 + 8 = 23;

б) помножимо одержану суму на 3: 23 × 3 = 69;

в) визначаємо суму цифр, які розташовані на непарних місцях кодового слова: 5 + 3 + 9 + 2 + 2 + 4 = 25;

г) визначаємо суму двох результатів за п. п. *б* та *в*:
69 + 25 = 84;

д) визначаємо контрольну цифру як різницю між числом кратним 10, що є найближчим більшим за одержаний результат у п. *г*, і числом одержаним у п. *г*: 90 – 84 = 6, тобто $u_К = 6$.

У коді EAN-13 штрихове зображення має дві частини, по 6 знаків у кожній, які розділені знаком *H*4, і має зліва і справа обмежувальні знаки *H*1 (СТАРТ і СТОП). За першою зліва цифрою визначаємо набори кодових комбінацій, якими кодуються цифри першої частини кодового слова (табл.6.3). У зв'язку з тим, що $u_{12} = 4$, згідно табл. 6.3 маємо: u_{11}, u_9 та u_8 – кодуються набором *A*, а u_{10}, u_7 та u_6 – набором *B*. Цифри $u_5, \ldots, u_1$, а також контрольна цифра $u_К$, кодуються відповідними знаками набору *C* (табл.6.1). Таким чином, кодове слово EAN-13 у двійковому еквіваленті має такий вигляд:

101 0110111 0011011 0011001 0010011 0011011 0010111 01010
*H*1 8 2 1 2 2 9 *H*4
1110010 1000010 1011100 1001110 1001000 1010000 101
0 3 4 5 8 6 *H*1.

Задача 2.

Зчитувальним пристроєм фіксується кодове слово у коді EAN-13: 4821223034586, у якому міститься помилка. Показати процес виявлення помилки.

Розв'язання. Для виявлення помилки у кодовому слові коду EAN-13 виконуємо перевірку на відповідність контрольної цифри ($u_K = 6$) цифрам кодового слова, що надійшло до декодера зчитувального пристрою. Для цього знаходимо контрольну цифру u_K^* для прийнятого кодового слова 482122303458 (без u_K) згідно з алгоритмом для коду EAN-13 та порівнюємо її з u_K :

$8 + 4 + 0 + 2 + 1 + 8 = 23;$

$23 \times 3 = 69;$

$5 + 3 + 3 + 2 + 2 + 4 = 19;$

$69 + 19 = 88;$

$90 - 88 = 2 \rightarrow u_K^* = 2, \quad u_K^* \neq u_K.$

Таким чином, контрольні цифри у прийнятому кодовому слові і обчислені декодером не збігаються. Це вказує на наявність помилки у прийнятому кодовому слові.

Тема 7. Лінійні блокові коди

ПЛАН

1. **Двійкові коди, що виявляють помилки**
2. **Двійкові коди, що виправляють однократні помилки**
3. **Циклічні коди**

1. ДВІЙКОВІ КОДИ, ЩО ВИЯВЛЯЮТЬ ПОМИЛКИ

Особливість кодів, що виявляють помилки, полягає у тому, що кодові комбінації, які входять до складу таких кодів, відрізняються одна від одної кодовою відстанню не меншою за $d_{min} = 2$.

Такі коди умовно можна розділити на дві групи: коди, в яких використовуються всі комбінації, але до кожної з них за обумовленим правилом додаються r перевірочних елементів, та коди, які одержують шляхом зменшення кількості дозволених комбінацій.

До першої групи кодів, що виявляють помилки, відносяться такі лінійні коди: з перевіркою на парність, з простим повторенням, інверсний (Бауера), кореляційний; нелінійні коди: з перевіркою на непарність, код Бергера. Прикладом коду другої групи є код з постійною вагою. Код з числом одиниць в комбінації, кратним трьом, може належати до першої або до другої групи кодів у залежності від методики його побудови.

Код з перевіркою на парність є найбільш поширеним кодом, який використовується для виявлення поодиноких помилок і всіх помилок непарної кратності. Код містить $(n-1)$ інформаційних та один перевірочний елемент і позначається як $(n, n-1)$ - код.

Перевірочний елемент визначається як сума за модулем 2 всіх інформаційних елементів: $b_1 = \sum_{i=1}^{k} \circ a_i$; тобто кодова комбінація коду утворюється доповненням комбінації k - елементного первинного коду одним елементом таким чином, щоб кількість одиниць у новому n - розрядному $(n = k+1)$ коді була парною. Код має кодову відстань $d_{\min} = 2$.

Для виявлення помилки на приймальному боці виконують перевірку на парність всієї прийнятої кодової комбінації за допомогою визначення кодового синдрому $s_1 = \sum_{i=1}^{n-1} \circ a_i^* \oplus b_1^*$;

де a_i^*, b_1^* – прийняті на приймальному боці відповідно інформаційні та перевірочний елементи.

Вважається, що при $s_1 = 0$ помилки в комбінації нема, при $s_1 = 1$ – помилка є. Код виявляє всі помилки непарної кратності.

Надмірність коду $R = 1 - k/(k+1) = 1/(k+1)$.

Код з перевіркою на непарність відрізняється від коду з перевіркою на парність тим, що кожна його кодова комбінація має непарне число одиниць, тобто додатковий перевірочний елемент формують виходячи з числа одиниць у первинній кодовій комбінації: при парному числі одиниць перевірочний елемент дорівнює одиниці, при непарному – нулю. Для виявлення помилки в кодовій комбінації на приймальному боці виконується перевірка на непарність. Код є роздільним нелінійним кодом довжини n з $n-1$ інформаційними та одним

перевірочним елементами і має таку ж спроможність вияв-лення помилки та надмірність, як і код з перевіркою на парність.

Код з простим повторенням (з повторенням без інверсії) є роздільним лінійним кодом. Код містить k інформаційних та $r = k$ перевірочних елементів. У цьому коді r перевірочних елементів є простим повторенням k інформаційних елементів первинної кодової комбінації: $b_i = a_i$, де $i = 1 \ldots k$. Через те, що код має $d_{min} = 2$, він може бути використаний для виявлення поодиноких помилок. Процедура виявлення помилок у прийнятій кодовій комбінації полягає у порівнянні однойменних інформаційних і перевірочних елементів. Їх незбіг говорить про наявність помилок у прийнятій комбінації. Код дозволяє виявити не тільки однократні помилки, а й деякі помилки більшої кратності, за винятком так званих "дзеркальних" помилок, коли в інформаційній і перевірочній послідовностях кодової комбінації в результаті дії завад спотворюються елементи, які знаходяться на однакових за номером розрядах.

Надмірність коду $R = 1 - k / (2 k) = 1 / 2.$

Інверсний код (код Бауера) є роздільним лінійним кодом з повторенням з інверсією, який має k інформаційних та k перевірочних елементів. Його відмінність від коду з простим повторенням полягає у тому, що значення перевірочних елементів у ньому залежать від значення суми за модулем 2 всіх інформаційних елементів. При $\sum_{i=1}^{k} \circ a_i = 0,$ тобто при парному числі одиниць у первинній кодовій комбінації перевірочні елементи просто повторюють інформаційні ($b_i = a_i$, де $i = 1 \ldots k$). При $\sum_{i=1}^{k} \circ a_i = 1,$ тобто при непарному числі одиниць у первинній кодовій комбінації, перевірочні елементи повторюють інформаційні в інвертованому вигляді (у зворотному коді): $b_i = a_i \oplus 1$, де $i = 1 \ldots k.$

Для виявлення помилок декодером у послідовності, що складається з $2k$ елементів, спочатку підсумовують одиниці, які знаходяться у перших k елементах. Якщо їх кількість парна, решта k елементів приймається у позитиві. Обидві зареєстровані частини кодової комбінації поелементно порівнюються (перший елемент з першим, другий – з другим і т.д.). При наявності хоча б одного незбігу вся послідовність елементів бракується. Якщо кількість одиниць серед перших k елементів прийнятої комбінації непарна, решта k елементів приймається у негативі (інвертуються). Після чого виконується поелементне порівняння. Наявність незбігу призводить до відбракування кодової комбінації. Така побудова коду дозволяє виявити дуже багато варіантів спотворення елементів.

Надмірність коду $R = 1 - k / (2 k) = 1 / 2.$

Кореляційний код передбачає кодування кожного елемента первинної кодової комбінації. При цьому "0" записується як "01", а "1" – як "10". Так, наприклад, первинній кодовій комбінації 100101 буде відповідати комбінація 100101100110 кореляційного коду. В технічній літературі такий двійковий запис дуже часто називають Манчестер - код. Приймальний пристрій на кожному такті, який складається з двох сусідніх елементів кореляційного коду, повинен зафіксувати перехід $0 \to 1$ або $1 \to 0$. У разі прийняття двох нулів або двох одиниць приймальний пристрій фіксує наявність помилки.

Такий код дозволяє виявляти помилки будь-якої кратності у кожній парі елементів одного такту, але не здатний виявити так звані "дзеркальні" двократні помилки, коли сусідні елементи одного такту під впливом завад змінюються на протилежні.

Надмірність коду $R = 1 - k/(2k) = 1/2$.

До переваг коду можна віднести, крім відсутності постійної складової у напрузі кодованого сигналу при передачі одиниць та нулів по каналу зв'язку імпульсами постійного струму різної полярності, також можливість самосинхронізації генератора приймача, тому що приймання кожного біта супроводжується фронтом сигналу, який приймається, у центрі біта.

Код Бергера є найбільш поширеним з несистематичних кодів. У такому коді перевірочні елементи, які дописуються у кінці первинної кодової комбінації, – це інвертований запис двійкового чис-ла, яким записується сума одиниць у кодовій комбінації k – елемент-ного первинного коду, що кодується кодом Бергера. При цьому число r перевірочних елементів визначається як найменше ціле, для якого виконуються умови $r \geq log_2(k+1)$. Так, наприклад, при $k = 8$, отри-маємо $log_2(8+1) = log_2 9 = 3{,}16993$, тобто $r = 4$.

Для виявлення помилки у декодері виконується операція підрахунку числа одиниць в інформаційній частині прийнятої кодової комбінації. Це число записується у двійковій формі, інвертується і порівнюється з перевірочною частиною прийнятої кодової комбінації. Їх незбіг вказує на наявність помилки.

Надмірність коду $R = 1 - r/n$.

Код з постійною вагою, тобто з постійним числом одиниць та нулів у комбінаціях, часто називають кодом на одне сполучення. Загальна кількість кодових комбінацій коду з постійною вагою

$$N = C_n^m = \frac{n!}{m!(n-m)!},$$

де m – число одиниць у комбінації довжини n.

Такий код утворюється з простого двійкового коду відбором комбінацій, які мають однакову кількість одиниць m. У декодері підраховується кількість одиниць у прийнятій кодовій комбінації. Невідповідність кількості одиниць числу m говорить про наявність помилки у кодовій комбінації.

Код з постійною вагою має мінімальну кодову відстань $d_{min} = 2$ і виявляє всі помилки непарної кратності, а також всі помилки пар-ної кратності, які призводять до порушення умови $m = const.$

Надмірність коду $R = 1 - (log_2 C_n^m)/n$.

Код з числом одиниць у комбінації, кратним трьом, можна утворити або шляхом додавання до кожної комбінації первинного коду двох перевірочних елементів, або зменшенням кількості дозволених кодових комбінацій первинного коду за допомогою накладання додаткової умови – кількість одиниць у кожній комбінації повинна бути кратною трьом (0, 3, 6, . . .)

У першому випадку до первинної кодової комбінації додаються два перевірочні розряди, які мають такі значення, що сума одиниць у кодовій комбінації стає кратною трьом. Так, наприклад, комбінація первинного коду 01100 закодована кодом з числом одиниць, кратним

трьом, буде мати вигляд 0110010, комбінація $01000 \rightarrow 0100011$, $1010 \rightarrow 101010$, $101100 \rightarrow 10110000$, $101110 \rightarrow 10111011, 0111110 \rightarrow 011111010$ тощо.

У другому випадку з усіх кодових комбінацій первинного коду вибирають тільки ті комбінації, які мають вагу $w = 0, 3, 6, 9, \ldots$. Всі інші комбінації заборонені для вживання.

Код дозволяє виявити всі поодинокі помилки та деякі помилки більшої кратності.

Здатність коду виявляти помилкові комбінації майже така ж, як і коду з постійною вагою.

Надмірність коду з доповненням до необхідної кількості одиниць (кратності): $R = 1 - k/(k+2)$, а для коду, який утворюється шляхом відбору комбінацій з відповідною кількістю одиниць з повного числа комбінацій простого коду:

$$R = 1 - [\, log_2 (C_n^0 + C_n^3 + C_n^6 + \ldots + C_n^{3b}) \,] / n ,$$

де b – ціла частина $n/3$.

2. ДВІЙКОВІ КОДИ, ЩО ВИПРАВЛЯЮТЬ ОДНОКРАТНІ ПОМИЛКИ

Коди, що виправляють одну помилку, повинні мати мінімальну кодову відстань $d_{min} \geq 3$. Збільшення кодової відстані досягається збільшенням числа розрядів коду n при незмінній кількості дозволених кодових комбінацій або зменшенням числа дозволених кодових комбінацій, що використовуються для передачі повідомлень, тобто збільшенням надмірності коду.

Найбільшого поширення серед двійкових кодів, що виправляють однократні помилки, одержали систематичні (лінійні, групові) блокові коди: Хеммінга, ітеративний (Елайеса), з багатократним повторенням, інверсний, та несистематичний код Бергера .

Систематичний код з $d_{min} = 3$, який називають кодом **Хем-мінга**, використовується для виправлення однієї або виявлення двох помилок.

Систематичний (груповий, лінійний) код довжиною n з кількістю інформаційних символів k позначають як (n, k) - код. Для систематичного (n, k) - коду з $d_{min} = 3$ кількість перевірочних символів вибирають як найменше ціле r, що відповідає умовам

$$2^r \geq n + 1 = k + r + 1. \qquad (8.1)$$

Як відомо, у систематичних кодах перевірочні елементи можна одержати шляхом додавання за модулем 2 визначених інформаційних елементів.

Систематичний код можна задавати *твірною* (*породжуваль-ною*) *матрицею*, якій притаманні такі особливості:

– матриця має k рядків та n стовпців ;

– кожний елемент матриці є або “0”, або “1”;

– кожний рядок матриці являє собою кодову комбінацію коду, що цією матрицею задається, і повинен мати не менше трьох одиниць ;

– всі рядки матриці повинні бути лінійно незалежними ;

– поелементна сума за модулем 2 будь - якої кількості рядків матриці (яка, до речі, завжди буде комбінацією коду) повинна мати не менше трьох одиниць.

Підібрані за даних умов вихідні комбінації, які називають *базисними*, записуються у вигляді матриці:

$$G_{n,k} = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1k} & b_{11} & b_{12} & \dots & b_{1r} \\ a_{21} & a_{22} & \dots & a_{2k} & b_{21} & b_{22} & \dots & b_{2r} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ a_{k1} & a_{k2} & \dots & a_{kk} & b_{k1} & b_{k2} & \dots & b_{kr} \end{bmatrix}, \qquad (8.2)$$

де a_{ji} та b_{jm} – відповідно i-ий інформаційний та m-ий перевіроч-ний елементи j-ої базисної кодової комбінацій.

Твірну матрицю (8.2) можна подати у вигляді двох підматриць: інформаційної (E_k) та перевірочної ($C_{r,k}$).

Інформаційну підматрицю зручно подати у канонічній формі як одиничну підматрицю розміром $k \times k$:

$$E_k = \begin{bmatrix} 1 & 0 & 1 & \dots & 0 \\ 0 & 1 & 0 & \dots & 0 \\ 0 & 0 & 1 & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & 1 \end{bmatrix}.$$

Перевірочна підматриця $C_{r,k}$ будується підбором r-розрядних двійкових послідовностей з числом одиниць у кожному рядку не менше за $d_{min} - 1 = 2$. При цьому необхідно враховувати, що сума за модулем 2 будь-яких рядків цієї підматриці не повинна мати менше за $d_{min} - 2 = 1$ одиниць, тобо однакові набори є неприпустимими.

Рядки у перевірочній підматриці можна міняти місцями. При цьому можна одержати декілька варіантів твірних матриць.

Твірна матриця дозволяє одержати всі кодові комбінації систематичного групового коду. Це досягається послідовним додаванням за модулем 2 рядків матриці у всіх можливих сполученнях (тобто першого і другого рядків матриці; першого і третього; першого і четвертого; . . . ; другого і третього; другого і четвертого; . . .; першого, другого і третього; першого, другого і четвертого; . . . , нарешті усіх k рядків). Нульова комбінація дописується окремо.

Опираючись на твірну, матрицю можна побудувати *перевірочну матрицю* $H_{n,r}$, яка налічує r рядків та n стовпців. Перевірочна матриця складається з двох підматриць: підматриці $D_{k,r}$, яка має k стовпців та r рядків, кожний рядок якої відповідає транспонованому стовпцю перевірочної підматриці $C_{r,k}$ твірної матриці $G_{n,k}$, та одиничної підматриці E_r розміром $r \times r$:

$$H_{n,r} = [\, D_{k,r}; E_r \,] = \begin{bmatrix} b_{11} & b_{21} & \dots & b_{k1} & 1 & 0 & 0 & \dots & 0 \\ b_{12} & b_{22} & \dots & b_{k2} & 0 & 1 & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ b_{1r} & b_{2r} & \dots & b_{kr} & 0 & 0 & 0 & \dots & 1 \end{bmatrix}. \quad (8.3)$$

Перевірочна матриця (8.3) дозволяє спростити операції кодування і декодування.

Запишемо довільну кодову комбінацію коду у вигляді

$$V = [\, a_1 a_2 a_3 \; . \; . \; . \; a_k \; b_1 b_2 b_3 \; . \; . \; . \; b_r \,],$$

де a_i та b_m – відповідно інформаційні та перевірочні елементи.

Позиції, які зайняті одиницями у i-ому рядку підматриці $D_{k,r}$, визначають ті інформаційні елементи, які повинні брати участь у формуванні i-ого перевірочного елемента b_i:

$$\begin{aligned} b_1 &= b_{11}a_1 \oplus b_{21}a_2 \oplus \ldots \oplus b_{k1}a_k, \\ b_2 &= b_{12}a_1 \oplus b_{22}a_2 \oplus \ldots \oplus b_{k2}a_k, \\ &\vdots \\ b_r &= b_{1r}a_1 \oplus b_{2r}a_2 \oplus \ldots \oplus b_{kr}a_k. \end{aligned} \qquad (8.4)$$

Існування співвідношень (8.4), що пов'язують інформаційні та перевірочні елементи кодової комбінації, дає можливість при деко-дуванні виявляти та виправляти помилки в кодових комбінаціях, які можуть з'являтися через спотворення елементів у двійковому каналі під час їх передачі. Аналізуючи результати перевірки цих співвідношень у прийнятій кодовій комбінації, можна отримати певну ін-формацію про помилки.

Позначимо кодову комбінацію, яка пройшла через двійковий канал та підлягає декодуванню,

$$V^* = [\, a_1^* a_2^* a_3^* \ldots a_k^* b_1^* b_2^* b_3^* \ldots b_r^* \,],$$

де a_i^* та b_m^* – відповідно інформаційні та перевірочні елементи кодової комбінації на виході каналу.

Для з'ясування питання, чи відповідає кодова комбінація V^* правилам побудови коду, отримаємо набір s_j, $j = 1,2,3,\ldots,r$:

$$s_j = b_{1j}a_1^* \oplus b_{2j}a_2^* \oplus b_{3j}a_3^* \oplus \ldots \oplus b_{kj}a_k^* \oplus b_j^*.$$

Кожний елемент s_j дає інформацію про те, задовольняють чи ні символи кодової кобінації V^* відповідному рівнянню системи (8.4).

Набір елементів ($s_1, \ldots, s_r$) називається кодовим ***синдромом*** або пізнавачем помилок.

Якщо синдром складається з одних нулів, кодова комбінація V^* є *дозволеною*, тобто задовольняє правилам побудови коду.Такий результат буде мати місце, якщо в кодовій комбінації немає помилок або конфігурація помилок є такою, що вона не може бути виявлена цим кодом. Присутність хоча б одного ненульового елемента в комбінації синдрому вказує на спотворення хоча б одного елемента у прийнятій кодовій комбінації.

Значення синдрому при однократній помилці у прийнятій кодовій комбінації збігаються із стовпцями перевірочної матриці. Порівнюючи кодовий синдром з стовпцями матриці $H_{n,r}$, можна знайти місце помилки у комбінації по їх збігу. У разі помилки виправляється той розряд кодової комбінації, який відповідає порядковому номеру стовпця матриці, що збігається з синдромом.

Вкорочені систематичні (групові) коди утворюються з повних кодів, при цьому одержують d_{min} не меншу, ніж у повного систематичного коду зі збереженням такої ж кількості перевірочних елементів, тобто $r = (n-i)-(k-i) = n-k$.

Одержання вкороченого коду грунтується на тому, що через те що з загального числа 2^k комбінацій первинного коду, які потрібно закодувати, наприклад, систематичним $(n-1,\ k-1)$-кодом, 2^{k-1} ком-бінацій починаються з "0", а 2^{k-1} – з "1", то і після кодування 2^{k-1} комбінацій систематичного групового (n, k)-коду також будуть починатися з "0".

Будемо ці 2^{k-1} комбінацій розглядати як новий груповий код. Тоді, якщо вони належать вихідному (n, k) - коду, d_{min} нового коду буде не меншою, ніж d_{min} вихідного. Нульовий символ на початку вихідної кодової комбінації зберігається і після кодування її груповим (n, k) - кодом. Зрозуміло, що він не впливає на утворення переві-рочних елементів групового коду і його можна відкинути. При цьому одержимо груповий $(n-1, k-1)$ - код, тобто код, що містить $n-1$ елементів, з яких $k-1$ інформаційних та r – перевірочних.

Вкорочений груповий код легко одержати з повного (n, k) - коду, який поданий у вигляді твірної матриці $G_{n,k}$ розміру $(k \times n)$ вигляду (8.2), шляхом виключення з матриці першого рядка і першого стовпця. У результаті цього одержимо твірну матрицю $G_{n-1,\,k-1}$ нового вкороченого коду розмірності $(k-1) \times (n-1)$, що утворює 2^{k-1} комбінацій вкороченого коду. Для одержання перевірочної матриці $H_{n-1,\,r}$ вкороченого коду досить виключити перший стовпець з матриці $H_{n,\,r}$ відповідного повного коду.

Після вкорочення на один символ групового $(n-1, k-1)$ - коду можна одержати вкорочений $(n-2, k-2)$ - код тощо.

Наведена вище інтерпретація коду Хеммінга грунтується на сучасних уявленнях про цей код як різновид лінійних кодів. Згідно з цією теорією можна побудувати не менше, ніж $n!/r!$ різноманітних лінійних кодів, що мають $d_{min} = 3$, тобто кодів Хеммінга. Кожен із цих кодів задається однією із перестановок стовпців перевірочної матриці. Всі ці коди еквівалентні за корегувальною здатністю; відрізняються вони розташуванням перевірочних символів, співвідношеннями, яким відповідають символи, та, звичайно, наборами кодових комбінацій.

Код, запропонований Р.В.Хеммінгом ще до формування теорії лінійних кодів, – це один із варіантів вищеозначених кодів, для якого перевірочна матриця будується так, що i - ий стовпець її є двійковим поданням числа i. За такою умовою кодовий синдром, у разі виникнення однократної помилки, буде двійковим поданням номера спотвореного розряду кодової комбінації. Перевірочні розряди в такому коді розташовані на позиціях з номерами $2^0, 2^1, 2^2, \ldots, 2^{r-1}$; перевірочні розряди розміщуються між інформаційними.. Будемо називати такий код *традиційним кодом Хеммінга.*

Розширений код Хеммінга використовується, головним чином, для виявлення помилок. Цей код має кодову відстань $d_{min} = 4$ і забезпечує виявлення одно-, дво- і трикратних помилок завдяки введенню додаткового перевірочного елемента b_0, який одержують за допомогою перевірки кодової комбінації коду Хеммінга на парність. При цьому перевірочний елемент, який розміщується, як правило, на початку кодової комбінації, дорівнює “0” при парній кількості одиниць у кодовій комбінації і “1” – при непарній.

Декодування розширеного коду Хеммінга виконують у зворотній послідовності: спершу виконують загальну перевірку прийнятої кодової комбінації на парність, а потім – перевірку кодової комбінації без b_0. При цьому можуть виникнути ситуації, які показані у таблиці 8.1.

Для одержання вкорочених кодів Хеммінга з $d_{min} = 3$ або 4 керуються правилами, що були викладені при побудові вкорочених систематичних кодів.

Таблиця 8.1

Кратність помилки	Перевірка на парність (b_0)	Кодовий синдром
Помилка відсутня	$= 0$	$= \vec{0}$
Однократна помилка	$= 1$	$\neq \vec{0}$
Двократна помилка	$= 0$	$\neq \vec{0}$
Трикратна помилка	$= 1$	$\neq = \vec{0}$

Код з багатократним повторенням (з повторенням без інверсії) є роздільним лінійним кодом. Код містить k інформаційних та mk перевірочних елементів, де m – число повторень первинної кодової комбінації. У цьому коді кожні k перевірочних елементів є просто повтореннями інформаційних елементів

$$b_j = b_{j+k} = b_{j+2k} = \ldots = b_{j+(m-1)k} = a_j, \quad j = 1 \ldots k .$$

Кодова відстань коду з багатократним повторенням $d_{min} = m + 1$, тому при $m \geq 2$ код здатен не тільки виявляти, але і виправляти помилки. Процедура виявлення помилок у прийнятій кодовій комбінації полягає у порівнянні однойменних інформаційних і перевірочних розрядів. Їх незбіг говорить про наявність помилок у прийнятій комбінації. При виправленні помилок у комбінації застосовується мажоритарний принцип виправлення для кожного інформаційного елемента, тобто "голосування за більшістю", коли за істинне значення приймається те, яке частіше зустрічається у цьому інформаційному і відповідних йому перевірочних елементах. Код дозволяє виправляти всі помилки кратності від 1 до цілої частини числа $m / 2$ та деякі помилки більш високої кратності у залежності від розміщення помилок у комбінації.

Надмірність коду $R = m / (m + 1)$.

Ітеративні коди (коди Елайеса), якщо вони орієнтовані на виправлення однократних помилок, являють собою, як правило, двомірні лінійні коди з кодуванням рядків і стовпців завадостійкими кодами з перевіркою на парність (див. розділ 7). Такі ітеративні коди мають мінімальну кодову відстань $d_{min} = 4$ і у режимі виправлення помилок дозволяють виправити будь-які однократні помилки і деякі помилки більшої кратності.

Рекомендується на практиці використовувати коди з числом перевірочних елементів 8, 9 та 16. Для коду з $r = 8$ використовують блок інформаційних елементів розмірами 3×4 (з $k_1 = 3$ рядками та $k_2 = 4$ стовпцями). При цьому число інформаційних елементів $k = k_1 \times k_2 = 3 \times 4 = 12$, число перевірочних – $r = 8$, $n = 20$. Для коду з $r = 9$ беруть $k = k_1 \times k_2 = 4 \times 4 = 16$, $n = 25$; для коду з $r = 16$: або $k = k_1 \times k_2 = 8 \times 7 = 56$, $n = 72$ або $k = k_1 \times k_2 = 7 \times 8 = 56$, $n = 72$.

При виправленні помилки у декодері визначають рядок і стовпець, для яких не виконуються умови парності. Спотворений інформаційний елемент, розташований на місці перетину рядка і стовпця, для яких не виконується перевірка на парність, інвертується.

Надмірність двомірних ітеративних кодів:

для $r = 8 \rightarrow R = r / n = 2 / 5$;

для $r = 9 \rightarrow R = 9/25$;

для $r = 16 \rightarrow R = 2/9$.

Несистематичний код Бергера є найбільш поширеним з несистематичних кодів. У такому коді перевірочні елементи, які дописуються у кінці первинної кодової комбінації, – це інвертований за-пис двійкового числа, яке дорівнює сумі вагів тих елементів інфор-маційної частини кодової комбінації, на яких розташовані одиниці. При цьому число r перевірочних елементів визначається як найменше ціле, яке задовольняє нерівності $r \geq log_2 \sum_{i=1}^{k} w_i$, де w_i – вага i - ого інформаційного елемента первинної кодової комбінації, яка кодується кодом Бергера, а ваги елементів первинної комбінації повинні приймати такі значення, починаючи з першого (старшого) розряду: 3, 5, 6, 7, 9, 10, 11, 12, 13, 14, 15, 17, 18 тощо, тобто всі числа, крім тих, які дорівнюють значенням числа 2 у будь-якій цілій додатній степені ($2^0, 2^1, 2^2, \ldots$). Так наприклад, при $k = 8$ маємо $log_2 \sum_{i=1}^{8} w_i$ $= log_2 (3 + 5 + 6 + 7 + 9 + 10 + 11 + 12) = log_2 63 = 5{,}977$, тобто $r = 6$. Таким чином перевірочна частина кодової комбінації буде мати 6 розрядів.

Для виправлення помилки в декодері підраховується сума S^* вагів тих інформаційних розрядів прийнятої кодової комбінації, на яких розташовані одиниці. Далі інвертуються перевірочні розряди прийнятої кодової комбінації; отримане двійкове число переводиться у десяткове і віднімається від обчисленої суми S^*. Якщо в інформаційній частині кодової комбінації є однократна помилка, то модуль різниці буде збігатись із вагою спотвореного розряду; для виправлення помилки відповідний інформаційний розряд потрібно інвертувати.

Надмірність коду $R = 1 - k/n = 1 - k/(k+r) = r/n$.

3. Циклічні коди

Подання кодових комбінацій у циклічних кодах виконують у вигляді поліномів від формальної змінної x, що дозволяє звести дії над кодовими комбінаціями до дій над поліномами. Так, сума двох двій-кових поліномів виконується додаванням за модулем 2 коефіцієнтів за рівних степенів змінної x. Наприклад, отримаємо суму за модулем 2 двох поліномів: $(x^4 \oplus x^3 \square\, x \oplus 1) \oplus (x^3 \oplus x^2 \oplus x) = x^4 \oplus x^2 \oplus 1$. Множення виконується за звичайними правилами множення степеневих функцій, але коефіцієнти однакових степенів додаються за модулем 2. Так,
$(x^4 \oplus x^3 \oplus x \oplus 1)(x \oplus 1) = x^5 \oplus x^4 \oplus x^2 \oplus x \oplus x^4 \oplus x^3 \oplus x \oplus 1 = = x^5 \oplus x^3 \oplus x^2 \oplus 1$.
Ділення також виконується як звичайне ділення поліномів, при цьому операція віднімання співпадає з операцією додавання $\oplus$. Наприклад,
$(x^5 \oplus x^3 \oplus x^2 \oplus 1) / (x \oplus 1) = x^4 \oplus x^3 \oplus x \oplus 1$.
До циклічних належать лінійні блокові (n, k) - коди, у яких циклічний зсув елементів будь-якої дозволеної комбінації призводить до виникнення також дозволеної комбінації, що належить до даного коду. Така циклічна перестановка з'являється завдяки помноженню полінома даної комбінації на x. Щоб степінь полінома не перевищував $n - 1$, член x^n замінюється одиницею.

Особливу роль в теорії циклічних кодів відіграють твірні поліноми, у якості яких звичайно використовуються незвідні поліноми та їх добутки.

Циклічні коди з $d_{min} = 3$. Розрізняють алгебраїчні та матричні способи побудови циклічних кодів. Існують три алгебраїчні способи побудови кодових комбінацій циклічного коду, які випливають з виразу

$$\frac{x^r Q(x)}{P(x)} = C(x) \oplus \frac{R(x)}{P(x)}, \qquad (9.1)$$

де r – кількість перевірочних розрядів у комбінації циклічного коду; $Q(x)$ – поліном первинної кодової комбінації; $P(x)$ – твірний полі-ном; $C(x)$ – частка від ділення того степеня, що і $Q(x)$; $R(x)$ – остача від ділення, яка має степінь, не більший за $r-1$. З виразу (9.1) можна одержати три способи побудови циклічного коду :

$F_1(x) = x^r Q(x) \oplus R(x)$;

$F_2(x) = C(x) P(x)$;

$F_3(x) = Q(x) P(x)$,

де $F(x)$ – комбінація циклічного коду.

Перші два способи дають один і той же роздільний циклічний код, тобто $F_1(x) = F_2(x)$, у якому розташування інформаційних і перевірочних елементів буде підпорядковано правилу: k старших роз-рядів комбінації – інформаційні, решта $n-k=r$ розрядів – пере-вірочні. Третій спосіб використовується для побудови нероздільного циклічного коду, де інформаційні і перевірочні елементи в комбінаціях не відокремлені одні від одних, що ускладнює процес декодування.

Деякі твірні поліноми для циклічних кодів наведені у табл. 9.1.

Таблиця 9.1

Кількість перевірочних елементів r	Твірний поліном $P(x)$	Двійковий запис полінома
3	$x^3 \square x \square 1$	1011
3	$x^3 \square x^2 \square 1$	1101
4	$x^4 \square x \square 1$	10011
4	$x^4 \square x^3 \square 1$	11001
5	$x^5 \square x^2 \square 1$	100101
5	$x^5 \square x^3 \square 1$	101001
5	$x^5 \square x^3 \square x^2 \square x \square 1$	101111
5	$x^5 \square x^4 \square x^2 \square x \square 1$	110111
6	$x^6 \square x^5 \square x^4 \square 1$	1110001
8	$x^8 \square x^7 \square x^6 \square x^5 \square x^2 \square x \square 1$	111100111
9	$x^9 \square x^5 \square x^3 \square 1$	1000101001

Очевидно, що $F(x)$ повинен ділитися на $P(x)$ без остачі. На цьому і грунтується перевірка кодової комбінації на наявність помилок при прийомі. Якщо прийнята комбінація $F^*(x)$ ділиться на $P(x)$ без остачі, вона визнається безпомилковою. Якщо після ділення залишається ненульова остача, це свідчить про наявність помилки у прийнятій комбінації циклічного коду.

Для виправлення помилки можна скористатися методом гіпотез. Цей метод грунтується на послідовній побудові гіпотез про помилки у молодшому розряді прийнятої кодової комбінації, потім, якщо гіпотеза не підтверджується, у другому розряді і так далі, поки гіпотеза не підтвердиться і остача від ділення $F^*(x) \oplus E(x)$, де $E(x)$ – поліном помилки, на $P(x)$ не дасть нульовий результат. Це означає, що $F(x) = F^*(x) \oplus E(x)$ і помилка виправлена.

Розрізняють три матричні способи одержання циклічного коду, два з яких грунтуються на побудові твірної матриці, і один – перевірочної.

За першим способом будується твірна матриця

$$G = \begin{bmatrix} x^0 \cdot P(x) \\ x^1 \cdot P(x) \\ x^2 \cdot P(x) \\ \vdots \\ x^{k-1} \cdot P(x) \end{bmatrix}.$$

За другим способом також будується твірна матриця циклічного коду, але на відміну від першого, у якості рядків такої матриці використовують усі можливі комбінації коду, що мають тільки одну одиницю в інформаційній частині; перевірочні елементи для таких комбінацій можна визначити за допомогою першого алгебраїчного способу побудови коду.

Третій матричний спосіб побудови циклічного коду грунтується на одержанні перевірочної матриці за допомогою використання перевірочного полінома, який визначається за формулою:

$H(x)=(x^n \square 1)/P(x)$,

де $n=2^r-1$. При цьому, перевірочна матриця має вигляд:

$$H = \begin{bmatrix} x^0 \cdot H(x) \\ x^1 \cdot H(x) \\ x^2 \cdot H(x) \\ \vdots \\ x^{r-1} \cdot H(x) \end{bmatrix}.$$

Процес кодування і декодування за допомогою твірної або перевірочної матриць циклічного коду провадиться аналогічно даному процесу у двійковому груповому коді, викладеному в розділі 8.

Циклічні коди з $d_{min} = 4$ можуть виявляти одно-, дво- і трикратні помилки. Для збільшення кодової відстані до $d_{min} = 4$ кількість перевірочних елементів у кодовій комбінації такого коду має бути на один більшою, ніж у коді з $d_{min} = 3$. Твірний поліном $P(x)_{(d=4)}$ такого коду визначається як добуток твірного поліному $P(x)_{(d=3)}$ циклічного коду, який має $d_{min} = 3$, на поліном $(x \oplus 1)$, тобто:

$P(x)_{(d=4)} = P(x)_{(d=3)}(x \square 1)$.

Процедура кодування і декодування залишається такою ж, як і для циклічного коду з $d_{min} = 3$.

Вкорочені циклічні коди будуються за аналогією з двійковими груповими кодами на основі побудови твірної або перевірочної матриць (див. розділ 8).

Коди Боуза-Чоудхурі-Хоквінгема (БЧХ) є різновидом циклічних кодів з кодовою відстанню $d_{min} \geq 3$. Коди БЧХ дозволяють виявляти і виправляти будь-яку кількість помилок у залежності від мінімальної кодової відстані. При кодуванні задаються кількістю помилок,

яку слід виправити, або кодовою відстанню і загальною кількістю елементів у кодовій комбінації n. Кількість інформаційних елементів k та перевірочних елементів r визначається при побудові коду.

Так довжина кодової комбінації n у кодах БЧХ визначається з виразів [24]:

$$n = 2^h - 1, \quad \text{або} \quad n = (2^h - 1)/g, \qquad (9.2)$$

де h – ціле число; g – непарне додатне число, при діленні на яке одержуємо n цілим непарним числом. Таким чином, n може бути тільки непарним числом. Тобто, керуючись виразом (9.2), визначаємо, що кількість елементів у комбінаціях коду БЧХ може дорівнювати 3, 7, 15, 31, 63, 127, 255, 511, 1023 тощо.

Кількість перевірочних елементів коду відповідає співвідношенню

$$r \le h(d_{min} - 1)/2, \qquad (9.3)$$

звідки кількість інформаційних елементів

$$k \ge (2^h - 1) - h(d_{min} - 1)/2. \qquad (9.4)$$

Твірний поліном коду БЧХ визначається як добуток так званих мінімальних поліномів $M_i(x)$, із непарними індексами:

$$P(x) = M_1(x)M_3(x)...M_V(x), \qquad (9.5)$$

де $V = d_{min} - 2$. Можна пересвідчитись, що кількість мінімальних поліномів у добутку (9.5) дорівнює максимальній кількості s помилок, які гарантовано виправляються кодом.

У таблиці 9.2 наведені основні параметри деяких кодів БЧХ.

Таблиця 9.2

n	k	r	d_{min}	Твірний поліном P_8
7	4	3	3	13
15	11	4	3	23
	7	8	5	721
	5	10	7	2467
31	26	5	3	45
	21	10	5	3551
	16	15	7	107657
	11	20	11	5423325
	6	25	15	313365047
63	57	6	3	103
	51	12	5	12471
	45	18	7	1701317
	39	24	9	166623567
	36	27	11	1033500423
	30	33	13	1574641656547
	24	39	15	17323260404441
	18	45	21	1363026512351725

Подані в таблиці параметри були визначені у відповідності з викладеною вище методикою. Для зручності запису твірні поліноми $P(x)$ подані у вісімковій системі числення

(P_8). Щоб одержати твірний поліном у звичайному вигляді, тобто у тій формі, яка використовується для побудови кодів БЧХ, треба перевести кожну вісімкову цифру у двійковий трибіт. Так, наприклад, $P_8 = 45$ запишеться двійковими числами: 4 – 100 та 5 – 101. Таким чином одержуємо двійкове число 100101, яке записується поліномом $P(x) = x^5 \square x^2 \square 1$.

Як було показано вище, коди БЧХ мають непарне значення мінімальної кодової відстані d_{min}. Для того, щоб збільшити d_{min} на одиницю, досить помножити твірний поліном коду БЧХ на двочлен ($x \square 1$).

Кодування у кодах БЧХ виконується так само, як і у звичайних циклічних кодах, у тому числі і за допомогою матричних способів (див. вище). Декодування кодів БЧХ (виявлення та виправлення помилок) також може виконуватися з використанням методики, викладеної для циклічних кодів з $d_{min} < 5$.

.

Тема 8. Недвійкові коди

Недвійкові коди за аналогією з двійковими можна поділити на такі: первинні коди; коди, що виявляють помилки; коди, що виправляють помилки.

Недвійкові первинні коди використовуються у телекомунікаційних системах та мережах і системах телемеханіки. Далі наведені вирази для розрахунку кількості N_0 кодових комбінацій, які можна отримати при побудові таких первинних кодів (вони пов'язані з відповідним розділом математики, який називається комбінаторикою).

***Код на перестановки*:**

$N_0 = q!,\ q = n$;

тут і далі q – потужність алфавіту коду, n – довжина кодової ком-бінації.

***Код на певне число розміщень*:**

$N_0 = A_q^n = q! / (q - n)!,\ q > n$.

***Код на певне число сполучень*:**

$N_0 = C_q^n = q! / [(q - n)!\, n!],\ q > n$.

Код на всі сполучення:

$N_0 = q^n,\ q \geq n$.

Змінно-якісний код:

$N_0 = q(q-1)^{n-1}$.

Недвійкові коди, що виявляють помилки, можуть бути побудовані або введенням додаткових перевірочних елементів, які одержують як результат операцій над елементами первинної кодової комбінації, або збільшенням надмірності за рахунок зменшення кількості дозволених кодових комбінацій коду. В обох випадках досягається збільшення кодової відстані до значення, що дозволяє виявити ту чи іншу кількість помилок у кодовій комбінації.

Код з перевіркою за mod q будується за аналогією з двійковим кодом з перевіркою на парність, але з тією різницею, що виконується доповнення кодової комбінації первинного q-ічного коду перевіроч-ним елементом таким чином, щоб сума усіх елементів дорівнювала нулю за $mod\, q$. Значення перевірочного елемента у даному разі визначається різницею між q та сумою значень всіх елементів первинної кодової комбінації за $mod\, q$.

Такий код має незначну надмірність $R = 1/(k+1)$ і дозволяє виявити наявність помилок у кодовій комбінації, якщо сума усіх елементів (інформаційних та перевірочного) за $mod\, q$ відрізняється від нуля.

Код з простим повторенням є аналогом двійкового коду з простим повторенням (див. розділ 7), в основу якого покладено просте повторення первинної кодової комбінації. Алгоритм побудови коду має вигляд:

$b_i = a_i,\ i \in [1, k]$,

де a_i – інформаційний елемент, що знаходиться на i-ій позиції інфор-маційної частини кодової комбінації; b_i – перевірочний елемент, що знаходиться на i-ій позиції перевірочної частини кодової комбінації; k – кількість інформаційних елементів.

Надмірність коду $R = 0{,}5$. Код дозволяє виявити всі помилки, за винятком деяких помилок на однакових позиціях в інформаційній та перевірочній частинах коду.

Незвідний змінно-позиційний код НЗЗПК задовольняє таким умовам:

кожна кодова комбінація містить однакову кількість елементів, які передаються послідовно;

кожний елемент кодової комбінації містить m позицій алфаві-ту потужністю q;

сусідні елементи кодової комбінації повинні відрізнятися хоча б однією позицією;

останній елемент кодової комбінації не може збігатися з першим елементом комбінації, тобто для першого і останнього елементів кодової комбінації вибирають різні багатопозиційні сполучення.

Виконання останньої умови забезпечує незвідність коду, що дозволяє виконувати передавання елементів коду без пауз і у деяких випадках відмовитися від синхронізації, а також спрощує процедуру декодування.

Послідовність побудови НЗЗПК:

береться m символів з q позицій алфавіту;

визначається кількість сполучень позицій за заданим числом позицій у кожному сполученні;

вся кількість сполучень позицій розбивається на n груп, де n – кількість елементів кодової комбінації (довжина коду), і кожна група взаємно однозначно закріплюється за елементом кодової комбінації;

утворюються кодові комбінації з n елементами за правилом: для кожного елемента беруться сполучення позицій із закріпленої за даним елементом групи сполучень позицій.

За методом побудови кодових комбінацій багатопозиційні НЗЗПК поділяють на такі класи:

без розділення алфавіту коду на групи;

з розділенням алфавіту коду на v груп, кожна з котрих має q_i символів.

Останній клас, у свою чергу, поділяють на НЗЗПК, кожний елемент якого містить m позицій, які беруться з різних груп сполучень, і НЗЗПК, кожний елемент якого містить m позицій, які беруться з однієї групи сполучень.

Кількість N_0 кодових комбінацій для *НЗЗПК без розділення алфавіту коду* на групи

– у разі, коли всі сполучення позицій розподілені між n групами сполучень порівну

$$N_0 = (C_q^m / n)^n;$$

– у разі, коли вся кількість сполучень розподіляється між групами сполучень не нарівно через остачу деякої кількості сполучень

$$N_0 = (C_q^m - Q)^{n-Q} \times (C_q^m - Q + n)^Q / n^n,$$

де Q – остача від ділення C_q^m / n, що є цілим додатним числом.

Для *НЗЗПК з розділенням алфавіту коду* на v груп з однако-вою кількістю $l = q / v$ позицій у кожній групі та за умов, що кожний кодовий елемент містить m позицій, які вибираються з різних груп ($v \geq m$) кількість кодових комбінацій

$N_0 = (C_v^m \; l^m / n)^n$;

якщо ж для кожного кодового елемента m різних позицій вибираються із однієї групи ($l \geq m, v = n$), то

$N_0 = (C_l^m)^n$.

Недвійкові коди, що виправляють помилки, поділяють на блокові і неперервні. Блокові q-коди, як і аналогічні двійкові, призначені для виправлення, головним чином, незалежних помилок. Однак, на відміну від двійкових, один елемент недвійкового коду несе

$log_2 q$ біт інформації у залежності від методу побудови конкретного коду. Така особливість недвійкових кодів дає підставу стверджувати, що блоковий недвійковий код дозволяє виправляти умовний пакет помилок з $log_2 q$ біт інформації, який, якби він виникнув у аналогічному двійковому коді, не міг бути виправлений ним. Це є однією з переваг використання недвійкових кодів, що виправляють помилки.

Код з багатократним повторенням застосовується при передачі інформації по каналам з високим рівнем завад. Цей код містить k інформаційних елементів, а кількість r перевірочних елементів залежить від числа m повторень, причому кожний перевірочний елемент збігається з відповідним йому інформаційним. Таким чином довжина кодової комбінації: $n = k + k\,m = k(1 + m)$. Алгоритм побудови коду:

$$b_i^j = a_i, \; i \in [1, k], \; j \in [1, m],$$

де a_i – інформаційний елемент, що знаходиться на i-ій позиції інфор-маційної частини кодової комбінації; b_i^j – перевірочний елемент, що знаходиться на i-ій позиції j-го повторення.

Процедура виявлення помилок у прийнятій кодовій комбінації полягає у порівнянні однойменних інформаційних і перевірочних розрядів. Їх незбіг говорить про наявність помилок у прийнятій комбінації. При виправленні помилок у комбінації застосовується мажоритарний принцип виправлення для кожного інформаційного елемента, тобто “голосування за більшістю”, коли за істинне значення приймається те, яке частіше зустрічається у цьому інформаційному і відповідних йому перевірочних елементах.

При m-кратному повторенні надмірність коду $R = m/(1 + m)$.

Код з простим повторенням та перевіркою за mod q є ко-мбінованим використанням двох недвійкових кодів, що виявляють помилки: з простим повторенням та перевіркою за $mod\ q$. Це дає можливість одержати код, що виправляє однократні помилки. В основу коду покладене просте повторення первинної кодової комбінації з подальшим доповненням кодової комбінації перевірочним елементом, який отримують як доповнення до суми елементів первинної кодової комбінації до значення потужності q алфавіту коду (основи коду). Значення перевірочного елемента визначається різницею між q та сумою значень всіх елементів первинної кодової комбінації за $mod\ q$.

Код дозволяє виправити тільки всі однократні помилки. При цьому місце помилки визначається в декодері простим співставленням інформаційної та повтореної частин прийнятої кодової комбінації. Незбіг значень елементів у деякому розряді вказує на помилку

у цьому розряді. Для визначення, в якій частині кодової комбінації виникла помилка: в інформаційній чи повтореній, виконують перевірку за $mod\, q$ кожної частини комбінації окремо. При цьому у перевірку за $mod\, q$ повинен входити і перевірочний елемент, який міститься у кінці кодової комбінації. При перевірці отримуємо контрольний елемент, значення якого при відсутності помилки буде дорівнювати нулю, а при виникненні однократної помилки – буде відрізнятися від нуля.

При виникненні помилки в інформаційній частині прийнятої кодової комбінації її виправляють. Виправлене значення інформа-ційного елемента отримують як різницю за $mod\, q$ між спотвореним інформаційним елементом та одержаним при перевірці у декодері контрольним елементом.

Надмірність коду $R = (k+1)/(2k+1)$.

Узагальнений код Хеммінга УКХ є найбільш поширеним з недвійкових кодів такого класу. Код призначений для виправлення однократних помилок у термінах недвійкової арифметики.

Перевірочна матриця УКХ має розмір $r \times n$:

$H = [h_{ij}]$, $i = 1, 2, 3, \ldots, r$; $j = 1, 2, 3, \ldots, n$;

де r – кількість перевірочних елементів, n – довжина кодової ком-бінації, i – номер рядка, j – номер стовпця. Стовпці матриці H повинні бути ненульовими і різними та, крім того, всі вони повинні мати однакову першу ненульову компоненту δ. Оскільки δ відповідає умові $1 \le \delta \le q-1$, число її можливих значень дорівнює $q-1$. Звідси, виключивши нульовий вектор-стовпець, отримаємо число вектор-стовпців у матриці H (отже і максимальну довжину n_{max} кодової комбінації)

$$n_{max} = (q^r - 1)/(q-1). \qquad (10.1)$$

При побудові матриці H виявляється, що r перших її вектор-стовпців, кожний з яких містить єдину ненульову компоненту δ, утворюють діагональну підматрицю розміру $r \times r$. Ця обставина вказує на зручні позиції для розміщення r перевірочних елементів у комбінації.

Загалом макет кодової комбінації X можна подати як:

$$X = b1\ b2\ b3 \ldots br\ a1a2\ a3 \ldots ak, \qquad (10.2)$$

де a_j, b_i – q-ічні відповідно інформаційні та перевірочні елементи кодової комбінації.

З розв'язання фундаментального матричного рівняння систематичних кодів

$H X^{\mathrm{T}} = \mathbf{0}$

відносно перевірочних елементів b_i, випливає

$$b_i = \delta^{-1} \sum_{j=1}^{k} a_j h_{i,j+r}, \qquad (10.3)$$

де X^{T} – транспонований вектор X.

Показане у (10.1) розміщення перевірочних елементів не є єдино можливим, але воно забезпечує мінімальний обсяг обчислень при кодуванні та декодуванні. При цьому вираз (10.2) дає оптимальний алгоритм кодування. Кодовий вектор (10.1) поелементно передається у канал зв'язку у вигляді певних сигналів, де він може бути спотворений завадою. Фізичні явища процесу спотворення сигналів не мають значення, тому що з алгебраїчних міркувань його доцільно подати як $Y = X \oplus E\ (mod\, q)$, де Y – спотворений кодовий вектор на виході

тракту передачі; E – вектор помилки з єдиною ненульовою компонентою e (у припущенні однієї помилки).

Процедура декодування містить у собі, як перший крок, за аналогією з двійковим кодом Хеммінга, обчислення r-компонентного перевірочного синдрому: $S = H\,Y^{\mathrm{T}} = L\,e$. Вектор S являє собою помножений на значення помилки e ($1 \le e \le q-1$) стовпець L перевірочної матриці H, що відповідає позиції спотвореного елемента у кодовому векторі Y. Його називають *локатором місця помилки*. Оскільки всі вектор-стовпці у H мають першу ненульову компоненту, що дорівнює δ, то у S перша ненульова компонента s_1 завжди буде $s_1 = \delta e$. Це обумовлює другий крок процедури декодування – визначення *значення помилки*:

$$e = s_1\delta^{-1} = s_1/\delta . \qquad (10.4)$$

З визначення $S = L\,e$ випливає третій крок процедури декодування – знаходження локатора місця помилки:

$$L = S/e , \qquad (10.5)$$

тобто кожна компонента вектора S ділится на значення помилки e.

На четвертому кроку процедури декодування шляхом упорядкованого перебору стовпців перевірочної матриці H і порівняння кожного з локатором L по збігу визначають позицію спотвореного елемента у кодовій комбінації. П’ятим і останнім кроком декодування є виправлення помилки, яке виконується відніманням за $mod\,q$ значення помилки e від значення спотвореного елемента, знайденого у Y за його локатором L. Після цього спотворений елемент у прийнятій кодовій комбінації замінюють результатом віднімання і, виключивши перевірочні елементи, подають одержувачу інформаційну частину кодової комбінації.

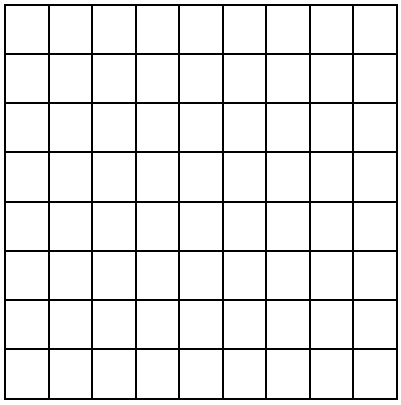
Надмірність коду $R = r/n$.

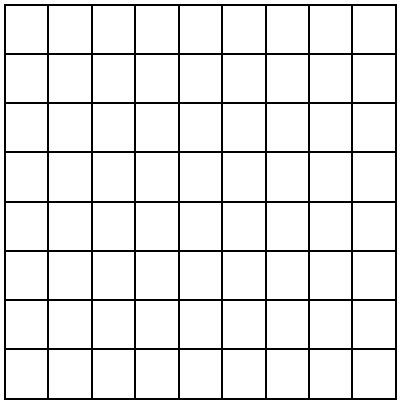
Зазначимо, що недвійкові коди прийнято ділити на дві великі групи: коди з простою основою, коли q є простим числом, і коди з основою q, що розкладається. Найбільший практичний інтерес має окремий випадок таких кодів при $q = 2^h$, символи якого мають інформаційну ємність h біт і можуть бути співставленні з усіма h-розрядними двійковими числами. Вибір q впливає на визначення операцій додавання, віднімання, множення і ділення, які використовуються у процедурах кодування і декодування. Якщо основа – просте число, зручно використати апарат обчислень за модулем простого числа. Якщо ж $q = 2^h$, то необхідно звернутися до алгебраїчного апарату обчислень за модулем незвідного полінома. Символи коду при цьому ставлять у відповідність елементам скінченого поля порядку q, а саме $GF(q)$.

У табл. 10.1.1 та 10.1.2 наведені результати операцій додавання та множення у скінченому полі $GF(2^3) = GF(8)$. Операцію додавання + двох елементів виконано як порозрядне додавання за $mod\,2$ їх двійкових еквівалентів з подальшим записом цього числа у вісімковому поданні. Операція множення × виконується як множення двох поліномів, що відповідають елементам поля, за модулем незвідного полінома степеня $h = 3$ (у даному разі $x^3 \oplus x \oplus 1$).

Таблиця 10.1.1

Таблиця 10.1.2





Для виконання операцій додавання і множення у скінченому полі $GF(16)$ зручніше користуватися адитивною та мультипліка-тивною формами запису десяткового числа, які подані у табл. 10.1.3. Операції виконуються на основі модульного полінома $P(x) = x^4 \oplus x \oplus 1$, який задає розширювальне рівняння $x^4 = x \oplus 1$ або $\beta^4 = \beta \oplus 1$. В першій колонці цієї таблиці наведені елементи поля $GF(16)$ у десятковому поданні (ці номерні записи можна розглядати як імена окремих елементів). У другій колонці наведено подання елементів поля $GF(16)$ у формі двійкового вектора (сукупності коефіцієнтів двійкового поліному – остачі від ділення на $P(x) = x^4 \oplus x \oplus 1$) з природнім розподілом вагів $2^3\ 2^2\ 2^1\ 2^0$. У третій колонці наведено мультиплікативну форму подання елементів поля $GF(16)$ у вигляді степенів примітивного елемента $\beta \in GF(16)$.

Можна замінити будь-який високий степінь β на такий, що не перевищує 14, якщо мати на увазі, що $\beta^{15} = 1$, $\beta^{16} = \beta$, $\beta^{17} = \beta^2$, $\beta^{18} = \beta^3$, . . .

Таблиця 10.1.3

Елементи $GF(16)$	Подання у формі вектора $2^3\ 2^2\ 2^1\ 2^0$	β^j	Адитивна форма $a\beta^3 \oplus b\beta^2 \oplus c\beta^1 \oplus d\beta^0$, $a,b,c,d \in \{0,1\}$ у базисі $\beta^3\ \beta^2\ \beta^1\ \beta^0$
0	0 0 0 0	$0 = 1/\beta^\infty$	0
1	0 0 0 1	$\beta^0 = 1$	1
2	0 0 1 0	β^1	β
3	0 0 1 1	β^4	$\beta \oplus 1$
4	0 1 0 0	β^2	β^2

5	0 1 0 1	β^8	$\beta^2 \oplus 1$
6	0 1 1 0	β^5	$\beta^2 \oplus \beta$
7	0 1 1 1	β^{10}	$\beta^2 \oplus \beta \oplus 1$
8	1 0 0 0	β^3	β^3
9	1 0 0 1	β^{14}	$\beta^3 \oplus 1$
10	1 0 1 0	β^9	$\beta^3 \oplus \beta$
11	1 0 1 1	β^7	$\beta^3 \oplus \beta \oplus 1$
12	1 1 0 0	β^6	$\beta^3 \oplus \beta^2$
13	1 1 0 1	β^{13}	$\beta^3 \oplus \beta^2 \oplus 1$
14	1 1 1 0	β^{11}	$\beta^3 \oplus \beta^2 \oplus \beta$
15	1 1 1 1	β^{12}	$\beta^3 \oplus \beta^2 \oplus \beta \oplus 1$

Виконаємо, наприклад, множення елементів 13 та 15 поля $GF(16)$ за допомогою степенів β:

$13 \times 15 = \beta^{13} \times \beta^{12} = \beta^{25} = \beta^{10} \times \beta^{15} = \beta^{10} \times 1 = \beta^{10} = 7.$

Код Ріда-Соломона (РС) застосовується для передачі інфор-мації по каналах з високою інтенсивністю завад, за яких виникають помилки кратністю два і більше та пачки помилок. Коди РС розглядають як такий випадок кодів БЧХ, коли поле локаторів збігається з полем його елементів. Тобто, якщо поле елементів $GF(q)$ БЧХ-коду має q окремих елементів (потужність поля q), а поле його локаторів $GF(q^l)$ має q^l елементів і є l-розширенням поля елементів $GF(q)$, то у РС-коді і елементи коду і їх локатори знаходяться у одному полі і належать $GF(q)$. Інакше кажучи, РС-код – це вироджена форма БЧХ-коду, у якого $l = 1$.

Довжина блока РС-коду $n = q - 1$, де q – потужність алфавіту (основа) коду.

РС-код, як і БЧХ-код, може задаватися твірною чи перевірочною матрицями або твірним чи перевірочним поліномами. Найбільш поширений спосіб побудови РС-коду на основі твірного поліному $P(x)$. Перевірочну матрицю H часто використовують для вивчення деяких властивостей РС-коду та його зв'язку з систематичними кодами.

Твірний поліном $P(x)$ коду РС, що виправляє s помилок, є добутком r мінімальних поліномів для спектру елементів з $GF(q)$, де $r = 2s$ – кількість перевірочних елементів у блоці коду:

$$P(x) = (x - \beta^{j})(x - \beta^{j+1})(x - \beta^{j+2})\ldots(x - \beta^{j+r-1}), \qquad (10.6)$$

де $\beta^{j}, \beta^{j+1}, \beta^{j+2}, \ldots, \beta^{j+r-1}$ – спектр твірних коренів поліному $P(x)$.

Степінь такого поліному дорівнює числу перевірочних елементів $r = 2s$.

Для спрощення побудови РС - коду часто вибирають $j = 1$ та одержують:

$$P(x) = (x - \beta^{1})(x - \beta^{2})\ldots(x - \beta^{r}) = \prod_{j=1}^{r}(x - \beta^{j}). \qquad (10.7)$$

Перевірочний поліном $H(x)$ РС - коду одержують як частку від ділення $(x^{q-1} - 1)$ на $P(x)$.

Мінімальна кодова відстань РС - коду $d_0 = r + 1 = 2s + 1$.

Надмірність коду $R = r/n = 2s/(q - 1)$.

Найбільш просто коди РС реалізуються для алфавіту потужністю $q = 2^{h}$, тобто коли $q = 4, 8, 16, \ldots$.У цьому випадку операції віднімання співпадають з операціями додавання, тому скрізь можна знак "$-$" замінити на "$+$", зокрема у виразах (10.6) та (10.7).

Декодування кодів РС виконується у відповідності з загальними принципами циклічних кодів . У разі виправлення помилок одержують значення локаторів $L_j = \beta^{j}$, що відповідають спотвореним елементам, і значення помилок для кожного спотвореного елемента. Виправлення помилок виконують відніманням від значення відповідного елемента значення помилки. Для полегшення виконання операцій результати додавання і множення у полі $GF(16)$ наведені табл. 10.3.

Недвійкові ітеративні коди мають більш високу здатність виявляти помилки у порівнянні з аналогічними двійковими ітератив-ними кодами. При $q > 2$ зростає обсяг інформації, що передається, завдяки тому, що кількість інформації, яка міститься в одному елементі кодової комбінації, визначається потужністю q алфавіту коду. У таких ітеративних кодах для кодування по рядках і стовпцях використовують недвійкові коди з перевіркою за $mod\, q$. Виявлення помилок виконується за аналогією з двійковим кодом – порівнянням перевірочних елементів кожного рядка (стовпця), поданих з каналу до декодера елементів коду, та одержаних у декодері шляхом обчислень.

Виправлення спотвореного елемента виконують наступним чином. Якщо не виконується перевірка для i-го рядка та j-го стовпця, то елемент, що знаходиться на перетинанні i-го рядка та j-го стовпця, замінюють елементом, який є сумою за $mod\, q$ даного прийнятого елемента (помилкового) та перевірочного елемента i-го рядка (або j-го стовпця), який був одержаний у декодері.

При виникненні декількох помилок у одному рядку (стовпці), помилки виправляють послідовно для тих стовпців (рядків), де вони є поодинокими.

степенів додаються за модулем 2. Так, $(x^4 \oplus x^3 \oplus x \oplus 1)(x \oplus 1) = x^5 \oplus x^4 \oplus x^2 \oplus x \oplus x^4 \oplus x^3 \oplus x \oplus 1 = = x^5 \oplus x^3 \oplus x^2 \oplus 1$.

Ділення також виконується як звичайне ділення поліномів, при цьому операція віднімання співпадає з операцією додавання $\oplus$. Наприклад, $(x^5 \oplus x^3 \oplus x^2 \oplus 1) / (x \oplus 1) = x^4 \oplus x^3 \oplus x \oplus 1$.

До циклічних належать лінійні блокові (n, k) - коди, у яких циклічний зсув елементів будь-якої дозволеної комбінації призводить до виникнення також дозволеної комбінації, що належить до даного коду. Така циклічна перестановка з'являється завдяки помноженню полінома даної комбінації на x. Щоб степінь полінома не перевищував $n-1$, член x^n замінюється одиницею.

Особливу роль в теорії циклічних кодів відіграють твірні поліноми, у якості яких звичайно використовуються незвідні поліноми та їх добутки.

Циклічні коди з $d_{min} = 3$. Розрізняють алгебраїчні та матричні способи побудови циклічних кодів. Існують три алгебраїчні способи побудови кодових комбінацій циклічного коду, які випливають з виразу

$$\frac{x^r Q(x)}{P(x)} = C(x) \oplus \frac{R(x)}{P(x)}, \qquad (9.1)$$

де r – кількість перевірочних розрядів у комбінації циклічного коду; $Q(x)$ – поліном первинної кодової комбінації; $P(x)$ – твірний полі-ном; $C(x)$ – частка від ділення того степеня, що і $Q(x)$; $R(x)$ – остача від ділення, яка має степінь, не більший за $r-1$. З виразу (9.1) можна одержати три способи побудови циклічного коду :

$F_1(x) = x^r Q(x) \oplus R(x)$;

$F_2(x) = C(x) P(x)$;

$F_3(x) = Q(x) P(x)$,

де $F(x)$ – комбінація циклічного коду.

Перші два способи дають один і той же роздільний циклічний код, тобто $F_1(x) = F_2(x)$, у якому розташування інформаційних і перевірочних елементів буде підпорядковано правилу: k старших роз-рядів комбінації – інформаційні, решта $n-k=r$ розрядів – пере-вірочні. Третій спосіб використовується для побудови нероздільного циклічного коду, де інформаційні і перевірочні елементи в комбінаціях не відокремлені одні від одних, що ускладнює процес декодування.

Деякі твірні поліноми для циклічних кодів наведені у табл. 9.1.

Таблиця 9.1

Кількість перевірочних елементів r	Твірний поліном $P(x)$	Двійковий запис полінома
3	$x^3 \square x \square 1$	1011
3	$x^3 \square x^2 \square 1$	1101
4	$x^4 \square x \square 1$	10011
4	$x^4 \square x^3 \square 1$	11001
5	$x^5 \square x^2 \square 1$	100101
5	$x^5 \square x^3 \square 1$	101001
5	$x^5 \square x^3 \square x^2 \square x \square 1$	101111

5	$x^5 \square x^4 \square x^2 \square x \square 1$	110111
6	$x^6 \square x^5 \square x^4 \square 1$	1110001
8	$x^8 \square x^7 \square x^6 \square x^5 \square x^2 \square x \square 1$	111100111
9	$x^9 \square x^5 \square x^3 \square 1$	1000101001

Очевидно, що $F(x)$ повинен ділитися на $P(x)$ без остачі. На цьому і грунтується перевірка кодової комбінації на наявність помилок при прийомі. Якщо прийнята комбінація $F^*(x)$ ділиться на $P(x)$ без остачі, вона визнається безпомилковою. Якщо після ділення залишається ненульова остача, це свідчить про наявність помилки у прийнятій комбінації циклічного коду.

Для виправлення помилки можна скористатися методом гіпотез. Цей метод грунтується на послідовній побудові гіпотез про помилки у молодшому розряді прийнятої кодової комбінації, потім, якщо гіпотеза не підтверджується, у другому розряді і так далі, поки гіпотеза не підтвердиться і остача від ділення $F^*(x) \oplus E(x)$, де $E(x)$ – поліном помилки, на $P(x)$ не дасть нульовий результат. Це означає, що $F(x) = F^*(x) \oplus E(x)$ і помилка виправлена.

Розрізняють три матричні способи одержання циклічного коду, два з яких грунтуються на побудові твірної матриці, і один – перевірочної.

За першим способом будується твірна матриця

$$G = \begin{bmatrix} x^0 \cdot P(x) \\ x^1 \cdot P(x) \\ x^2 \cdot P(x) \\ \vdots \\ x^{k-1} \cdot P(x) \end{bmatrix}.$$

За другим способом також будується твірна матриця циклічного коду, але на відміну від першого, у якості рядків такої матриці використовують усі можливі комбінації коду, що мають тільки одну одиницю в інформаційній частині; перевірочні елементи для таких комбінацій можна визначити за допомогою першого алгебраїчного способу побудови коду.

Третій матричний спосіб побудови циклічного коду грунтується на одержанні перевірочної матриці за допомогою використання перевірочного полінома, який визначається за формулою:

$H(x)=(x^n \square 1)/P(x)$,

де $n=2^r-1$. При цьому, перевірочна матриця має вигляд:

$$H = \begin{bmatrix} x^0 \cdot H(x) \\ x^1 \cdot H(x) \\ x^2 \cdot H(x) \\ \vdots \\ x^{r-1} \cdot H(x) \end{bmatrix}.$$

Процес кодування і декодування за допомогою твірної або перевірочної матриць циклічного коду провадиться аналогічно даному процесу у двійковому груповому коді, викладеному в розділі 8.

Циклічні коди з $d_{min} = 4$ можуть виявляти одно-, дво- і трикратні помилки. Для збільшення кодової відстані до $d_{min} = 4$ кількість перевірочних елементів у кодовій комбінації такого коду має бути на один більшою, ніж у коді з $d_{min} = 3$. Твірний поліном $P(x)_{(d=4)}$ такого коду визначається як добуток твірного поліному $P(x)_{(d=3)}$ циклічного коду, який має $d_{min} = 3$, на поліном $(x \oplus 1)$, тобто :

$P(x)_{(d=4)} = P(x)_{(d=3)}(x \square 1)$.

Процедура кодування і декодування залишається такою ж, як і для циклічного коду з $d_{min} = 3$.

Вкорочені циклічні коди будуються за аналогією з двійковими груповими кодами на основі побудови твірної або перевірочної матриць (див. розділ 8).

Коди Боуза-Чоудхурі-Хоквінгема (БЧХ) є різновидом циклічних кодів з кодовою відстанню $d_{min} \geq 3$. Коди БЧХ дозволяють виявляти і виправляти будь-яку кількість помилок у залежності від мінімальної кодової відстані. При кодуванні задаються кількістю помилок, яку слід виправити, або кодовою відстанню і загальною кількістю елементів у кодовій комбінації n. Кількість інформаційних елементів k та перевірочних елементів r визначається при побудові коду.

Так довжина кодової комбінації n у кодах БЧХ визначається з виразів [24]:

$$n = 2^h - 1, \quad \text{або} \quad n = (2^h - 1)/g, \qquad (9.2)$$

де h – ціле число; g – непарне додатне число, при діленні на яке одержуємо n цілим непарним числом. Таким чином, n може бути тільки непарним числом. Тобто, керуючись виразом (9.2), визначаємо, що кількість елементів у комбінаціях коду БЧХ може дорівнювати 3, 7, 15, 31, 63, 127, 255, 511, 1023 тощо.

Кількість перевірочних елементів коду відповідає співвідношенню

$$r \leq h(d_{min} - 1)/2, \qquad (9.3)$$

звідки кількість інформаційних елементів

$$k \geq (2^h - 1) - h(d_{min} - 1)/2. \qquad (9.4)$$

Твірний поліном коду БЧХ визначається як добуток так званих мінімальних поліномів $M_i(x)$, із непарними індексами:

$$P(x) = M_1(x)M_3(x)...M_V(x), \qquad (9.5)$$

де $V = d_{min} - 2$. Можна пересвідчитись, що кількість мінімальних поліномів у добутку (9.5) дорівнює максимальній кількості s помилок, які гарантовано виправляються кодом.

У таблиці 9.2 наведені основні параметри деяких кодів БЧХ.

Таблиця 9.2

n	k	r	d_{min}	Твірний поліном P_8
7	4	3	3	13
15	11	4	3	23
	7	8	5	721
	5	10	7	2467
31	26	5	3	45
	21	10	5	3551
	16	15	7	107657
	11	20	11	5423325
	6	25	15	313365047
63	57	6	3	103
	51	12	5	12471
	45	18	7	1701317

39	24	9	166623567
36	27	11	1033500423
30	33	13	1574641656547
24	39	15	17323260404441
18	45	21	1363026512351725

Подані в таблиці параметри були визначені у відповідності з викладеною вище методикою. Для зручності запису твірні поліноми $P(x)$ подані у вісімковій системі числення (P_8). Щоб одержати твірний поліном у звичайному вигляді, тобто у тій формі, яка використовується для побудови кодів БЧХ, треба перевести кожну вісімкову цифру у двійковий трибіт. Так, наприклад, $P_8 = 45$ запишеться двійковими числами: 4 – 100 та 5 – 101. Таким чином одержуємо двійкове число 100101, яке записується поліномом $P(x) = x^5 \square x^2 \square 1$.

Як було показано вище, коди БЧХ мають непарне значення мінімальної кодової відстані d_{min}. Для того, щоб збільшити d_{min} на одиницю, досить помножити твірний поліном коду БЧХ на двочлен $(x \square 1)$.

Кодування у кодах БЧХ виконується так само, як і у звичайних циклічних кодах, у тому числі і за допомогою матричних способів (див. вище). Декодування кодів БЧХ (виявлення та виправлення помилок) також може виконуватися з використанням методики, викладеної для циклічних кодів з $d_{min} < 5$.

.

Тема 9. Методи стиснення даних без втрат інформації

1. Поняття стиснення даних

Стиснення даних - це процес перетворення інформації у таку форму, яка потребує меншого обсягу пам'яті для зберігання або меншої кількості бітів для передавання каналами зв'язку. Його головна мета полягає у зменшенні обсягу подання даних без втрати їх корисного змісту або, залежно від методу, з допустимою втратою окремих характеристик. У межах цієї теми розглядається саме той випадок, коли інформація після відновлення має бути повністю тотожною початковому повідомленню.

У сучасних інформаційних системах проблема стиснення є надзвичайно актуальною. Це пов'язано з постійним зростанням обсягів цифрової інформації - текстових документів, програмних файлів, журналів подій, баз даних, конфігурацій, резервних копій, наукових даних, телеметрії, мережевих дампів та інших ресурсів. Навіть за стрімкого розвитку апаратних засобів зберігання й передавання даних потреба в економному використанні пам'яті та пропускної здатності не зникає, а навпаки посилюється.

Стиснення даних ґрунтується на тому, що більшість реальних повідомлень містять надлишковість. Це означає, що деякі частини повідомлення можна описати коротше без втрати змісту. Наприклад, у тексті окремі символи з'являються значно частіше за інші, у таблицях часто повторюються однакові значення, у програмному коді багато схожих конструкцій, а в цифрових зображеннях можуть бути великі області однакового кольору. Якщо таку повторюваність або передбачуваність виявити, її можна використати для компактнішого представлення інформації.

З інженерного погляду стиснення виконує кілька важливих функцій. По-перше, воно дозволяє економити дисковий простір. По-друге, зменшує час передавання файлів мережею. По-третє, дає змогу ефективніше організувати резервне копіювання, архівування та зберігання інформації. По-четверте, у деяких випадках стиснення сприяє підвищенню продуктивності системи, оскільки менший обсяг даних потребує менших витрат на пересилання чи запис.

Водночас стиснення не є універсальним «зменшувачем» будь-яких даних. Його результативність залежить від властивостей самих даних. Якщо інформація вже має дуже низьку надлишковість, наприклад є зашифрованою або вже стисненою іншим методом, то подальше стиснення може бути малоефективним або взагалі неможливим.

1.1. Основна ідея стиснення

Основна ідея стиснення полягає в усуненні або зменшенні надлишковості у представленні даних. Якщо два різні способи запису несуть одну й ту саму інформацію, то більш короткий із них є кращим з погляду економії ресурсів. Таким чином, стиснення - це не зміна змісту повідомлення, а оптимізація форми його подання.

Наприклад, якщо в потоці часто повторюється один і той самий символ або одна й та сама послідовність символів, тоді немає потреби кожного разу записувати її повністю. Можна зафіксувати це повторення у скороченому вигляді. Саме на цій ідеї базується значна частина практичних алгоритмів.

1.2. Стиснення як частина життєвого циклу даних

У реальних інформаційних системах стиснення нерідко використовується на різних етапах життєвого циклу даних. Воно може застосовуватися під час створення файлу, збереження в архів, передавання мережею, зберігання в хмарі, резервного копіювання та навіть у процесі внутрішньої обробки даних програмою. Це означає, що стиснення є не лише окремою технічною процедурою, а й важливим компонентом архітектури цифрових систем.

Для майбутніх фахівців у галузі інформаційних систем важливо розуміти, що правильний вибір методу стиснення впливає на продуктивність сервісу, навантаження на мережу, швидкість доступу до даних і витрати на зберігання.

2. Класифікація методів стиснення

Методи стиснення можна класифікувати за кількома ознаками. Найбільш загальним є поділ на стиснення без втрат та стиснення з втратами. Саме цей поділ є базовим для розуміння

всієї теми, оскільки він визначає, чи можна після декомпресії повністю відновити початкову інформацію.

Стиснення без втрат означає, що процес є повністю оборотним. Після розпакування отримуємо точну копію оригінального файлу, повідомлення чи структури даних. Усі символи, байти, службові ознаки та логічні елементи залишаються незмінними. Такі методи використовуються для текстових документів, програмного коду, архівів, конфігураційних файлів, баз даних, електронних таблиць, журналів системних подій та інших даних, де точність є обов'язковою.

Стиснення з втратами допускає спрощення або часткове відкидання окремих даних. Після декомпресії відновлене повідомлення не є повністю тотожним оригіналу, але з практичного погляду може залишатися достатньо якісним. Такі методи характерні для мультимедійної інформації - зображень, аудіо, відео, де незначні втрати сприймаються як припустимі.

Окрім цього, методи стиснення можна класифікувати за принципом роботи. У такому випадку зазвичай виділяють:

- статистичні методи;
- словникові методи;
- серійні методи;
- комбіновані методи.

Статистичні методи ґрунтуються на ймовірностях появи символів і намагаються кодувати часті елементи коротше, а рідкісні довше. Словникові методи орієнтуються на повторювані фрагменти рядків або байтових послідовностей. Серійні методи добре працюють із довгими послідовностями однакових значень. Комбіновані алгоритми поєднують кілька підходів, щоб досягти кращого результату.

2.1. Поділ за сферою застосування

Методи стиснення також доцільно класифікувати за сферою використання. Одні алгоритми орієнтовані на текстові файли, інші - на графіку, ще інші - на потоки телеметрії або службові дані мережевих протоколів. Наприклад, RLE добре працює на простих графічних масивах з довгими серіями однакових значень, а словникові методи - на текстових документах і коді програм.

Такий поділ важливий тому, що один і той самий алгоритм може бути ефективним для одного типу даних і майже не давати виграшу для іншого. Тому під час проєктування інформаційної системи стиснення слід розглядати не абстрактно, а з урахуванням конкретного типу інформації.

2.2. Поділ за способом організації обробки

Методи стиснення можна розрізняти і за способом організації обробки даних. Деякі з них працюють потоково, тобто здатні стискати інформацію в міру її надходження. Інші потребують аналізу всього файлу або хоча б великого блоку. Потокові алгоритми зручні для мережевого передавання та реального часу. Блокові - часто забезпечують кращий результат, але можуть потребувати більше пам'яті й часу.

Для студента важливо розуміти, що класифікація методів - це не просто перелік назв. Вона допомагає побачити, який саме тип надлишковості використовує той чи інший алгоритм і в яких умовах його доцільно застосовувати.

3. Стиснення без втрат і його особливості

Стиснення без втрат інформації - це таке перетворення даних, після якого початкове повідомлення може бути відновлене повністю, без жодної зміни. Ця властивість є ключовою. Якщо хоча б один символ, байт або структурний елемент буде змінено, такий метод уже не можна вважати безвтратним.

Саме тому безвтратне стиснення є критично важливим у тих сферах, де дані мають точне логічне значення. Наприклад, у текстовому документі зміна навіть однієї літери може спотворити зміст. У вихідному коді програми один неправильний символ здатний зробити файл непридатним до компіляції. У фінансовій таблиці зміна цифри вплине на результат

обчислень. У базі даних одна помилка може порушити цілісність запису. У таких випадках втрати неприпустимі.
Особливість безвтратного стиснення полягає в тому, що воно не «відкидає» інформацію, а лише знаходить ефективніший спосіб її представлення. Це означає, що алгоритм повинен бути строго оборотним. Процес кодування і декодування тут є взаємно однозначними: кожному стислому запису має відповідати лише одне початкове повідомлення.

3.1. Чому безвтратне стиснення є важливим

Безвтратне стиснення лежить в основі багатьох цифрових сервісів і технологій. Архіватори, резервне копіювання, передавання конфігураційних даних, стиснення логів, робота з документами, файловими форматами й базами даних - усі ці задачі вимагають повного збереження інформації. Втрата навіть дрібної частини даних у таких системах може мати серйозні наслідки.

Крім цього, безвтратне стиснення часто використовується як допоміжний механізм у складних програмно-апаратних рішеннях. Воно знижує навантаження на канали зв'язку, зменшує обсяг резервних копій та дозволяє компактніше зберігати службові дані без порушення їх структури.

3.2. Що впливає на ефективність безвтратного стиснення

Ефективність безвтратного стиснення визначається тим, наскільки багато надлишковості міститься у вхідних даних. Якщо файл має повторювані символи, підрядки, слова, шаблони, однакові блоки або нерівномірний розподіл частот символів, то алгоритм зазвичай зможе добре його стиснути.

Якщо ж дані мають майже випадкову структуру, ефективність різко знижується. Це характерно, наприклад, для зашифрованих файлів, стиснених архівів або випадкових двійкових послідовностей. Такі дані вже мають низьку надлишковість, а тому майже не піддаються подальшому стисканню.
Отже, безвтратне стиснення не гарантує однакового результату для всіх типів файлів. Його можливості завжди обмежені реальною структурою інформації.

3.3. Основні вимоги до алгоритмів безвтратного стиснення

До алгоритмів безвтратного стиснення висувають кілька важливих вимог. По-перше, вони повинні забезпечувати точне відновлення даних. По-друге, бажано, щоб вони працювали достатньо швидко. По-третє, важливо, щоб обсяг службової інформації не був надто великим, інакше виграш від стиснення зменшиться. По-четверте, алгоритм має бути придатним для конкретного класу задач - архівування, мережевої передачі, резервного копіювання або вбудованих систем.
Таким чином, безвтратне стиснення - це не лише питання «чи можна зменшити файл», а й питання точності, швидкості, ресурсоємності та придатності до конкретного застосування.

4. Теоретичні основи безвтратного стиснення

Теоретичною основою безвтратного стиснення є поняття інформаційної надлишковості. Надлишковість означає, що повідомлення містить елементи, які можна передбачити, скоротити або описати компактніше без втрати змісту. Іншими словами, якщо дані мають закономірності, ці закономірності можна використати для побудови коротшого коду.

Одним із ключових понять теорії інформації є ентропія. Вона характеризує середню кількість інформації, яка припадає на один символ джерела повідомлень. Якщо символи джерела з'являються з дуже різними ймовірностями, ентропія зазвичай нижча, а можливості стиснення - вищі. Якщо ж усі символи майже рівноймовірні і зв'язки між ними слабкі, ентропія зростає, а потенціал стиснення зменшується.
З практичного погляду це означає, що добре стискаються ті повідомлення, які мають передбачувану структуру. Наприклад, у природній мові деякі літери, склади, слова та граматичні конструкції трапляються частіше за інші. У програмному коді часто повторюються

ключові слова, оператори та шаблони синтаксису. У службових файлах - типові теги, параметри й службові позначення. Усе це і є джерелом надлишковості.

4.1. Ентропія як межа стиснення

Ентропія вказує на теоретичну межу середнього кодування символів. Вона показує, наскільки сильно в принципі можна стиснути дані без втрат. Жоден алгоритм не здатен «обійти» цю межу для довільного джерела інформації. Якщо повідомлення вже майже не містить надлишковості, простір для додаткового стиснення буде мінімальним.

4.2. Типи надлишковості у даних

У даних можуть існувати різні види надлишковості. Найпростішою є частотна надлишковість, коли одні символи з'являються значно частіше за інші. Саме її використовують статистичні методи.

Іншим видом є структурна або шаблонна надлишковість, коли в потоці повторюються не окремі символи, а цілі фрагменти. Таку надлишковість використовують словникові алгоритми.

Ще один тип - локальна серійна надлишковість, коли однакові елементи йдуть поспіль довгими ланцюжками. На ній базується RLE.

Таким чином, різні алгоритми безвтратного стиснення орієнтовані на різні форми надлишковості. Саме тому в теорії і практиці виникає багато різних методів, кожен із яких по-своєму описує та використовує закономірності в даних.

4.3. Кодування як заміна однієї форми представлення іншою

Із теоретичного погляду стиснення - це особливий випадок кодування. Повідомлення не змінюється за змістом, але отримує нову форму представлення. Якщо ця нова форма є коротшою, маємо стиснення. Якщо її можна однозначно перетворити назад у початкову, то стиснення є безвтратним.

Тут важливо усвідомити, що стиснення - це не «зникнення» даних. Інформація не пропадає, а упаковується в компактніший вигляд за рахунок використання знань про її структуру.

4.4. Практичний зміст теоретичних основ

Теоретичні основи потрібні не лише для формальних визначень. Вони пояснюють, чому один метод добре стискає текст, інший - графіку, а третій - службові потоки. Розуміння ролі ентропії, надлишковості та закономірностей дозволяє правильно обирати метод стиснення в реальних інформаційних системах.

Для майбутнього фахівця це означає, що під час проєктування програмного рішення не варто обирати алгоритм «за звичкою». Потрібно оцінити характер даних, швидкість доступу, вимоги до точності, обсяг ресурсів і лише після цього визначати, який підхід буде найраціональнішим

5. Методи серійного кодування

Методи серійного кодування ґрунтуються на виявленні у вхідних даних послідовностей однакових символів, які повторюються підряд. Такі послідовності називають серіями або «ланцюжками повторів». Якщо в повідомленні є багато однакових елементів, записувати кожен із них окремо нераціонально, тому замість повного повторення використовують коротший опис виду «значення + кількість повторень».

Основна ідея цього підходу полягає в тому, що надлишковість може виникати не лише через загальну статистичну нерівномірність символів, а й через локальні повтори. Наприклад, якщо в масиві даних поспіль ідуть десять однакових байтів, їх можна подати значно коротше, ніж у вихідному вигляді. Саме на такому принципі базується серійне кодування.

Ці методи належать до найпростіших у реалізації. Вони не потребують складної математичної моделі, статистичного аналізу ймовірностей або побудови словника великого обсягу. У найпростішому випадку алгоритм лише переглядає потік даних зліва направо, визначає довжину серії однакових елементів і замінює її компактним описом.

5.1. Метод RLE

Найвідомішим представником серійного кодування є метод RLE - «Run-Length Encoding», тобто «кодування довжин серій». Його принцип дуже простий: якщо певний символ повторюється кілька разів поспіль, замість самого повторюваного фрагмента записують пару, що містить значення символу і число його повторень.
Наприклад, рядок
«AAAAAAABBBCC»
можна подати як
«7A3B2C»

У такому поданні довгі серії однакових символів займають значно менше місця, ніж початковий запис.
З погляду програмної реалізації алгоритм працює так:

- зчитується перший символ;
- ведеться підрахунок, скільки разів він повторюється поспіль;
- коли зустрічається інший символ, попередня серія фіксується у стислому вигляді;
- далі процес повторюється для наступного символу.

У двійкових, текстових або байтових даних це означає, що алгоритм аналізує потік значень і формує стислий потік пар «лічильник + символ» або «символ + лічильник», залежно від конкретної реалізації.

5.2. Переваги серійного кодування

Головною перевагою RLE є його надзвичайна простота. Алгоритм легко реалізувати навіть у малоресурсних системах, мікроконтролерах або вбудованих пристроях, де складні статистичні чи словникові методи є надто витратними. Він не потребує складних структур даних, великих таблиць чи значного обсягу оперативної пам'яті.

Ще однією важливою перевагою є висока швидкість. Оскільки алгоритм виконує лише послідовний перегляд даних і лічбу повторів, кодування та декодування відбуваються дуже швидко. Це робить метод придатним для потокової обробки, простих форматів файлів і спеціалізованих службових структур.
Також RLE добре працює в тих випадках, коли дані мають природні довгі серії однакових значень. Наприклад, у деяких растрових зображеннях великі ділянки фону можуть складатися з однакових кольорів. Аналогічно, в телеметрії або службових масивах можуть виникати довгі фрагменти нулів чи повторюваних кодів стану.

5.3. Недоліки серійного кодування

Попри простоту, метод RLE має істотне обмеження - він ефективний лише тоді, коли у вхідних даних справді є довгі серії однакових символів. Якщо ж повтори короткі або майже відсутні, стислий запис може виявитися не меншим, а іноді навіть більшим за початковий.

Наприклад, для тексту, в якому символи чергуються без довгих повторів, запис на кшталт «1А1Б1В1Г...» не дає виграшу. Навпаки, до кожного символу доведеться додавати службову інформацію про кількість повторів, що збільшить обсяг.

Отже, ефективність RLE сильно залежить від структури вхідного потоку. Для одних типів даних він надзвичайно вдалий, а для інших практично марний.

5.4. Сфери застосування RLE

Метод RLE найчастіше застосовується:

- у простих графічних форматах, де є великі області одного кольору;
- у монохромних або палітрових зображеннях;
- у факсимільних повідомленнях;
- у деяких форматах зберігання карт, схем і піктограм;
- у телеметричних масивах із повторюваними значеннями;
- у службових структурах, де часто виникають серії нулів чи однакових байтів.

У сучасних універсальних архіваторах RLE рідко використовується як єдиний механізм стиснення, але дуже часто виступає як допоміжний етап попередньої обробки перед застосуванням складніших алгоритмів.

5.5. Практичне значення серійного кодування

Для студентів важливо розуміти, що RLE - це не просто «примітивний» метод, а важливий приклад того, як надлишковість у даних може мати локальний характер. Він добре показує, що стиснення не завжди вимагає складної математики. Іноді достатньо правильно виявити повторювану структуру й компактно її описати.

Крім того, на прикладі RLE легко пояснити загальний принцип безвтратного стиснення: якщо вхідне повідомлення має закономірності, його можна переподати коротше, не втрачаючи змісту. Саме з такого базового розуміння зручно переходити до складніших словникових і статистичних методів.

6. Словникові методи стиснення

Словникові методи стиснення займають особливе місце серед алгоритмів безвтратного кодування. На відміну від статистичних методів, які працюють переважно з частотами окремих символів, словникові алгоритми орієнтуються на повторювані фрагменти рядків або послідовностей байтів. Основна ідея полягає в тому, що повторюваний фрагмент не потрібно кожного разу записувати повністю - достатньо один раз його зафіксувати, а далі посилатися на нього скороченим способом.

Це дуже важливо для текстових, програмних, табличних і службових даних, де часто зустрічаються однакові слова, частини слів, теги, ключові конструкції, заголовки, шаблони та структурні фрагменти. Якщо система здатна виявляти такі повтори, то обсяг подання даних може бути значно зменшений.

Словникові методи вважаються одними з найефективніших для широкого класу практичних файлів. Вони стали основою багатьох архіваторів, форматів передавання та засобів зменшення обсягу інформації у файлових і мережевих системах.

6.1. Загальна ідея словникового підходу

Словниковий підхід полягає в тому, що під час обробки потоку даних алгоритм формує або використовує набір уже відомих фрагментів. Такий набір називають словником. Коли в потоці знову зустрічається фрагмент, який уже був зафіксований раніше, алгоритм не записує його повністю, а замінює коротшим посиланням - наприклад індексом у словнику, зсувом, довжиною або іншою службовою структурою.

Це дозволяє ефективно стискати дані з повторюваними шаблонами. На відміну від RLE, де шукаються лише серії однакових символів, словникові методи виявляють складніші повтори - слова, підрядки, фрази, фрагменти команд, структури розмітки, конфігураційні ланцюжки тощо.

6.2. Алгоритм LZ77

Алгоритм LZ77 є одним із базових словникових методів. Його запропонували Абрахам Лемпель і Якоб Зів у 1977 році. Головна особливість цього алгоритму полягає в тому, що він не створює окремого зовнішнього словника у звичному розумінні, а використовує частину вже обробленого потоку як «історію» для пошуку збігів.

Алгоритм працює з ковзним вікном, яке умовно поділяють на дві частини:

- буфер уже прочитаних даних;
- буфер попереднього перегляду, тобто даних, які ще потрібно закодувати.

Поточний фрагмент порівнюється з тими даними, які вже були оброблені. Якщо в буфері історії знаходиться найдовший збіг, замість самого фрагмента записується спеціальний дескриптор. У типовому випадку він містить:

- зсув назад;
- довжину знайденого збігу;
- наступний символ після збігу.

Наприклад, якщо певний фрагмент тексту вже траплявся кількома символами раніше, алгоритм не записуватиме його повторно, а лише вкаже, де саме він був і якої він довжини.

Перевагою LZ77 є те, що він добре стискає дані з повторюваними шаблонами і не вимагає зберігати великий статичний словник. Словник тут фактично утворюється

автоматично з уже переданих даних. Це робить алгоритм гнучким і придатним для потокового стиснення.
Разом з тим ефективність LZ77 залежить від розміру вікна. Якщо вікно замале, алгоритм може не знайти далеких повторів. Якщо ж вікно велике, збільшується складність пошуку та обчислювальні витрати.

6.3. Алгоритм LZ78

Алгоритм LZ78 є наступним етапом розвитку словникових методів. На відміну від LZ77, він будує окремий словник фрагментів у процесі роботи. Спочатку словник порожній або містить лише базові символи. Під час аналізу потоку алгоритм шукає найдовший фрагмент, який уже є у словнику, а далі додає до нього новий символ і заносить новоутворену послідовність як новий елемент словника.

У результаті в потоці передаються не самі фрагменти, а пари виду:

- індекс уже відомого фрагмента;
- наступний символ.

Такий підхід дозволяє поступово накопичувати знання про структуру повідомлення. Якщо в тексті або потоці є повторювані конструкції, словник швидко «навчається» і далі все частіше замінює довгі фрагменти короткими індексами.
Перевагою LZ78 є логічна чіткість і компактне представлення повторюваних підрядків. Однак зі зростанням словника можуть збільшуватися витрати пам’яті, а керування словником стає окремою важливою задачею.

6.4. Алгоритм LZW

Алгоритм LZW - це вдосконалений варіант LZ78, який набув великого практичного значення. Його основна ідея полягає в тому, що словник на початку вже містить усі базові символи алфавіту, а далі розширюється новими фрагментами в процесі кодування. На відміну від LZ78, у LZW немає потреби щоразу явно передавати «новий символ» окремо - часто достатньо передавати лише код словникового елемента.

Це робить алгоритм більш компактним і ефективним у багатьох практичних випадках. LZW добре працює для текстових файлів, коду програм, таблиць, форматованих документів і деяких типів графіки. Саме цей алгоритм став основою для відомих форматів, зокрема GIF, а також певних засобів архівування.

Ще однією перевагою LZW є відносно проста декомпресія. Якщо декодер використовує ті самі правила формування словника, що й кодер, то словник відновлюється синхронно, без необхідності передавати його окремо разом із файлом.

6.5. Переваги словникових методів

Словникові методи мають низку важливих переваг. Насамперед вони ефективно використовують повторюваність фрагментів, а не лише окремих символів. Це робить їх особливо корисними для текстів, HTML- і XML-документів, вихідного коду, логів, конфігураційних файлів, таблиць і багатьох інших структурованих даних.

Ще однією перевагою є універсальність. На відміну від RLE, який ефективний лише для довгих серій однакових символів, словникові алгоритми можуть знаходити більш складні закономірності. Наприклад, якщо одна й та сама фраза повторюється в різних частинах документа, словниковий метод зможе це використати.

Також словникові алгоритми добре комбінуються з іншими підходами. Дуже часто після виявлення повторів результати додатково стискаються статистичним методом, наприклад кодуванням Хаффмана. Саме так будуються багато сучасних форматів архівування.

6.6. Недоліки словникових методів

Головним недоліком словникових методів є вища складність порівняно з RLE. Потрібно підтримувати структуру словника, виконувати пошук збігів, визначати найдовші підрядки, керувати обмеженим обсягом пам’яті та забезпечувати коректне відновлення даних під час декомпресії.

Також ефективність словникового стиснення залежить від характеру даних. Якщо вхідний потік має мало повторюваних фрагментів або є майже випадковим, виграш буде невеликим. Уже стиснені, зашифровані або псевдовипадкові дані зазвичай погано піддаються такому стисненню.

Ще одним обмеженням є потреба в пошуку збігів, що може вимагати значних обчислювальних ресурсів. У швидкісних або ресурсно обмежених системах це іноді стає критичним фактором.

7.7. Практичне застосування словникових методів

Словникові методи широко використовуються:

- в архіваторах;
- у форматах стислих файлів;
- у веб-технологіях;
- у системах резервного копіювання;
- у файлових службах;
- у текстових і табличних даних;
- у форматах зображень без втрат;
- у каналах передавання структурованої інформації.

Вони особливо цінні для інформаційних систем, тому що велика частина службових, текстових та конфігураційних даних має повторювану структуру. Це означає, що словникове стиснення допомагає зменшити розмір резервних копій, скоротити мережевий трафік і ефективніше використовувати дисковий простір.

8. Практичне застосування безвтратного стиснення

Методи безвтратного стиснення мають надзвичайно широке практичне застосування, оскільки в сучасних інформаційних системах дуже часто виникає потреба зменшити обсяг даних без будь-якого спотворення змісту. На відміну від мультимедійних методів із втратами, безвтратне стиснення використовується там, де точність відновлення інформації є обов’язковою умовою коректної роботи системи.

Найперше безвтратне стиснення асоціюється з архівуванням файлів. Архіватори дозволяють зменшувати розмір документів, текстових масивів, програмного коду, таблиць, логів, конфігурацій і цілих каталогів. Це важливо не лише для економії місця на носіях, а й для полегшення копіювання, передавання та резервного зберігання даних. Коли великі обсяги інформації стискаються перед архівацією, це зменшує навантаження на дискову підсистему і прискорює операції резервного копіювання.

У мережевих інформаційних системах безвтратне стиснення використовується для скорочення обсягу трафіку. Якщо текстові, структуровані або службові дані передавати у стислому вигляді, це дозволяє зменшити час передавання, скоротити навантаження на канали зв’язку та підвищити ефективність взаємодії між вузлами системи. Особливо важливо це для серверних застосунків, хмарних платформ, веб-служб, віддалених API, систем синхронізації та корпоративних мереж.

Для веб-технологій безвтратне стиснення має окреме значення. Веб-сторінки, таблиці стилів, сценарії, JSON-відповіді, XML-документи та інші текстові ресурси можуть передаватися у стислому вигляді. Це зменшує час завантаження сторінки, прискорює взаємодію користувача з сервісом і знижує обсяг переданих даних. З технічного погляду це сприяє кращій масштабованості системи, особливо за великої кількості запитів.

У системах керування базами даних безвтратне стиснення застосовується для таблиць, індексів, резервних копій, журналів транзакцій та архівних записів. У великих інформаційних системах це дозволяє раціональніше використовувати дисковий простір і знижувати витрати на зберігання. Крім того, у деяких випадках менший фізичний обсяг даних дає змогу пришвидшити читання з носія, хоча це залежить від балансу між витратами на декомпресію та виграшем від меншого обсягу.

У сфері резервного копіювання безвтратне стиснення має особливе значення. Резервна копія повинна відновлювати дані абсолютно точно, тому будь-які втрати неприпустимі.

Стиснення дозволяє зменшити розмір резервних архівів, оптимізувати використання сховищ і спростити передавання копій на віддалені сервери. Для корпоративних систем, де щодня формуються великі резервні набори, це має значний економічний ефект.

Ще однією важливою сферою є зберігання програмного коду, бібліотек і виконуваних файлів. Для таких даних безвтратне стиснення є єдино допустимим, оскільки навіть найменша зміна байта може зробити програму непрацездатною. Аналогічна ситуація спостерігається у сфері технічної документації, конфігураційних файлів, шаблонів розгортання, скриптів автоматизації та системних журналів.

У графічних форматах безвтратне стиснення також відіграє важливу роль. Воно застосовується там, де важлива точність кожного пікселя, наприклад у технічних схемах, документації, медичних зображеннях, архівних копіях креслень, скриншотах інтерфейсів, наукових ілюстраціях та інших матеріалах, де спотворення є небажаними.

8.1. Значення для інформаційних систем

В інформаційних системах безвтратне стиснення підвищує ефективність роботи інфраструктури. Воно допомагає зменшити розмір службових даних, пришвидшити передачу великих масивів інформації, знизити навантаження на сховища і поліпшити організацію резервування. У багатьох випадках це не допоміжна функція, а важливий компонент архітектури системи.

8.2. Значення для мережевих сервісів

У мережевих середовищах виграш від стиснення може бути особливо відчутним. Менший розмір повідомлень означає менше використання пропускної здатності каналу. Це важливо для віддалених сервісів, мобільних застосунків, телеметричних систем, API-запитів та обміну між мікросервісами.

9. Переваги методів безвтратного стиснення

Методи безвтратного стиснення мають низку суттєвих переваг, завдяки яким вони стали невід'ємною частиною сучасних цифрових технологій. Головною перевагою є повне збереження початкової інформації. Після декомпресії користувач або система отримує точну копію початкових даних без будь-яких змін. Це особливо важливо для текстів, таблиць, баз даних, програмного коду, службових конфігурацій та архівних ресурсів.

Другою важливою перевагою є універсальність. Безвтратне стиснення придатне для широкого спектра даних, якщо вони мають достатню надлишковість. Його можна застосовувати до документів, логів, таблиць, конфігурацій, системних журналів, пакетів телеметрії, баз даних і багатьох інших структурованих наборів.

Третьою перевагою є економія ресурсів зберігання. Менший обсяг файлів означає ефективніше використання дисків, серверних сховищ, хмарних ресурсів і резервних носіїв. Для організацій, які працюють з великими обсягами архівних або службових даних, це може мати істотне економічне значення.

Четвертою перевагою є зменшення навантаження на канали передавання даних. Якщо повідомлення має менший розмір, його передавання займає менше часу і споживає менше пропускної здатності. У мережевих сервісах це сприяє підвищенню швидкодії та зменшенню затримок.

П'ятою перевагою є можливість інтеграції в автоматизовані процеси. Багато архіваторів, протоколів обміну, серверних платформ, інструментів резервування і баз даних уже мають вбудовані засоби безвтратного стиснення. Це означає, що відповідні алгоритми можна використовувати як стандартний інструмент цифрової інфраструктури.

Ще однією перевагою є надійність з точки зору логічної цілісності даних. Якщо система використовує безвтратне стиснення, вона не ризикує втратити значущі символи, службові байти або структурні елементи. Для професійної діяльності це критично, оскільки правильність обробки інформації часто залежить від точності кожного фрагмента.

9.1. Точність як головна перевага

Серед усіх переваг безвтратного стиснення саме точність є найважливішою. У ряді прикладних сфер вона є не просто бажаною, а обов'язковою. Без повної точності неможливо коректно відновити програму, документ, таблицю чи службову конфігурацію.

9.2. Економічний ефект

Стиснення даних має і прямий економічний ефект. Воно дозволяє зменшити потребу у фізичному просторі зберігання, скоротити витрати на резервні сховища, знизити трафік у мережах та зменшити час виконання окремих операцій передавання. Для великих систем навіть відносно невелике зменшення обсягу даних у масштабі тисяч або мільйонів файлів дає відчутний результат.

9.3. Технологічна гнучкість

Безвтратні алгоритми можуть використовуватися окремо або в комбінації. Це дозволяє вибирати метод залежно від конкретного типу даних і технічних вимог. Саме тому безвтратне стиснення залишається актуальним для різноманітних цифрових платформ.

10. Недоліки та обмеження

Попри значні переваги, безвтратне стиснення має і свої обмеження. Насамперед воно не може забезпечити настільки високий коефіцієнт зменшення обсягу, як методи з втратами. Причина очевидна - безвтратний підхід не має права відкидати жоден елемент інформації. Усе, що міститься в початкових даних, повинно бути збережене у тому чи іншому вигляді.

Другим важливим обмеженням є сильна залежність ефективності від структури вхідних даних. Якщо файл містить багато повторів, статистичної нерівномірності або шаблонів, алгоритм працюватиме добре. Якщо ж дані мають майже випадковий характер, виграш може бути мінімальним або взагалі відсутнім. Це характерно для зашифрованих даних, уже стиснених архівів, деяких двійкових потоків та псевдовипадкових масивів.

Третім недоліком є додаткове обчислювальне навантаження. Стиснення і декомпресія вимагають ресурсів процесора, пам'яті та часу. Для потужних серверів це може бути незначною проблемою, але в системах реального часу, мобільних пристроях, вбудованих рішеннях або середовищах з високим навантаженням цей фактор потрібно враховувати дуже уважно.

Четвертим обмеженням є необхідність службової інформації. Багато алгоритмів разом зі стислими даними змушені зберігати словники, таблиці частот, коди, дескриптори повторів або інші допоміжні структури. Якщо сам файл невеликий або має слабку надлишковість, то службова інформація може зменшити виграш або повністю його нівелювати.

Ще одним аспектом є складність окремих алгоритмів. Наприклад, деякі методи є простими й швидкими, але дають посередній результат. Інші забезпечують кращий коефіцієнт стиснення, проте потребують складнішої реалізації та більшої обчислювальної потужності. Таким чином, у кожному конкретному випадку доводиться шукати компроміс між ефективністю, швидкодією та складністю.

Для практичної роботи важливо розуміти й те, що повторне стиснення вже стиснених даних зазвичай не має сенсу. Якщо файл уже пройшов ефективну компресію, додаткове стиснення рідко дає позитивний результат. У деяких випадках розмір може навіть збільшитися через появу додаткової службової інформації.

Безвтратне стиснення завжди обмежене реальною надлишковістю даних. Якщо інформація майже не містить передбачуваних структур, алгоритм не зможе істотно скоротити її обсяг. Це фундаментальне обмеження, яке випливає з теорії інформації.

У прикладних системах важливо враховувати, що зменшення обсягу даних часто досягається ціною додаткових обчислень. Тому вибір алгоритму повинен враховувати не лише коефіцієнт стиснення, а й швидкість кодування, швидкість декодування та вимоги до пам'яті.

Уже стиснені, зашифровані або випадковоподібні дані погано піддаються безвтратному стисненню. Це потрібно враховувати під час проєктування систем обміну, резервування і зберігання. Не кожен файл доцільно стискати повторно.

11. Порівняння основних безвтратних методів

Порівняння безвтратних методів стиснення дає змогу краще зрозуміти, у чому полягає специфіка кожного підходу і чому не існує одного універсального алгоритму для всіх випадків. Кожен метод орієнтований на певний тип надлишковості й тому найкраще працює лише за відповідних умов.

Метод RLE є найпростішим. Він добре стискає дані, що містять довгі серії однакових символів або байтів. Його переваги - швидкість, простота та малі вимоги до ресурсів. Проте він майже неефективний для звичайних текстів, коду програм або даних без довгих повторів. Отже, RLE доцільно розглядати як спеціалізований або допоміжний метод.

Статистичні методи, зокрема Хаффман і арифметичне кодування, працюють із частотами появи символів. Вони корисні тоді, коли деякі елементи з'являються значно частіше за інші. Кодування Хаффмана є відносно простим і широко використовується як компонент складніших алгоритмів. Арифметичне кодування зазвичай забезпечує кращу ефективність, але потребує складнішої реалізації. Отже, статистичні методи добре підходять для даних із нерівномірним розподілом символів.

Словникові методи, такі як LZ77, LZ78 і LZW, орієнтовані на повторювані фрагменти рядків, а не лише окремі символи. Саме тому вони особливо добре працюють на текстових документах, конфігураційних файлах, вихідному коді, структурованих таблицях, логах та подібних даних. Їхня сила полягає в умінні використовувати повтори цілих шаблонів. Недоліком є більша складність і потреба в організації пошуку збігів або підтримці словника.

Комбіновані методи поєднують переваги кількох підходів. Наприклад, алгоритм може спочатку виявляти повторювані фрагменти за словниковим принципом, а потім додатково стискати результат статистичним методом. Саме такі схеми часто використовуються на практиці, оскільки дозволяють досягти кращого балансу між коефіцієнтом стиснення, швидкодією та універсальністю.

11.1. Порівняння за принципом роботи

RLE працює з локальними серіями однакових символів.
Хаффман і арифметичне кодування працюють із частотними закономірностями.
LZ-методи працюють із повторюваними підрядками та шаблонами.
Комбіновані алгоритми поєднують кілька різних типів аналізу надлишковості.

Таке порівняння показує, що різні алгоритми «бачать» різні форми структури в даних. Саме це визначає їх сильні та слабкі сторони.

11.2. Порівняння за складністю реалізації

RLE є найпростішим для реалізації.
Хаффман є помірно складним і добре придатним для практичного використання.
Арифметичне кодування складніше в реалізації й обчисленнях.
LZ77, LZ78 і LZW потребують словникових структур та пошуку збігів.
Комбіновані алгоритми зазвичай є найбільш складними, але й найбільш практично корисними.

Для майбутнього фахівця це означає, що вибір алгоритму завжди залежить не лише від бажаного коефіцієнта стиснення, а й від ресурсних можливостей системи.

11.3. Порівняння за сферою доцільного використання

RLE доцільний для простих графічних або службових даних із серіями повторів.
Статистичні методи доцільні для текстів і потоків з нерівномірною частотою символів.
Словникові методи є особливо корисними для текстів, конфігурацій, коду і структурованих документів.

Комбіновані підходи найкраще підходять для універсальних архіваторів і багатофункціональних систем стиснення.

Тема 10. Архівація даних

1. Поняття архівації даних

Архівація даних - це процес упорядкування одного або кількох файлів і папок у спеціальний архівний файл, який може зберігати інформацію у компактнішому, зручнішому для передавання або зберігання вигляді. У найзагальнішому розумінні архів є цифровим контейнером, у якому об'єднуються різні об'єкти файлової системи. Це можуть бути текстові документи, таблиці, графічні файли, програмні модулі, каталоги проєктів, службові журнали, резервні копії та інші дані.

Архівація є важливою тому, що вона дає змогу не просто «скласти» файли в одне місце, а й упорядкувати їх для подальшого використання. Якщо користувач має десятки чи сотні окремих документів, набагато зручніше передавати або зберігати їх у вигляді одного архівного файлу. Це знижує ймовірність втрати окремих елементів, полегшує структурування інформації та робить роботу з великими наборами даних більш керованою.

У багатьох випадках архівація поєднується зі стисненням. Це означає, що архіватор не лише створює контейнер, а й аналізує вміст і намагається подати його в компактнішій формі. Проте варто розуміти, що архівація не завжди обов'язково означає сильне зменшення розміру. Якщо дані вже стиснені або мають низьку надлишковість, архів буде виконувати передусім функцію упаковки, а не суттєвого скорочення обсягу.

Для сучасних інформаційних систем архівація є базовою операцією. Вона використовується в освітньому середовищі, офісній роботі, системному адмініструванні, хмарних сервісах, розробці програмного забезпечення, резервуванні та довготривалому зберіганні даних. Саме тому розуміння архівації є необхідною складовою цифрової грамотності фахівця з інформаційних технологій.

2. Призначення архівації в інформаційних системах

Архівація в інформаційних системах виконує одразу кілька важливих функцій. Найперше її призначення полягає у зручному об'єднанні великої кількості файлів у єдиний логічний об'єкт. Наприклад, навчальний проєкт може містити текст пояснювальної записки, презентацію, програмний код, зображення, таблиці та додаткові матеріали. Передавати все це окремо незручно, тоді як архів дозволяє зібрати ці об'єкти в один файл.

Друге важливе призначення архівації - зменшення загального обсягу даних. У разі, коли файли містять достатню надлишковість, архіватор може скоротити їх розмір. Це дає змогу заощадити місце на жорсткому диску, флеш-накопичувачі, сервері або в хмарному сховищі. Для великих організацій, які постійно працюють із масивами документів, звітів, конфігурацій і журналів подій, така економія може бути суттєвою.

Третя функція пов'язана з передаванням інформації. Надсилати один архівний файл значно простіше, ніж десятки окремих вкладень. Це особливо актуально для електронної пошти, навчальних платформ, внутрішніх корпоративних систем, веб-сервісів обміну файлами та систем дистанційної роботи. Крім того, архівація зменшує ризик того, що частина матеріалів буде пропущена або втрачена під час пересилання.

Четверте призначення архівації полягає в упорядкуванні та каталогізації інформації. Архіви часто використовуються для довготривалого зберігання завершених проєктів, навчальних матеріалів за певний рік, старих версій документів, резервних комплектів даних і технічної документації. У такому вигляді інформацію легше ідентифікувати, класифікувати та переносити між різними носіями.

П'яте призначення стосується часткового захисту даних. Багато архіваторів дозволяють встановити пароль на відкриття архіву або шифрувати його вміст. Це не замінює комплексну систему інформаційної безпеки, проте дає додатковий рівень захисту під час передавання або зберігання конфіденційних матеріалів.

Отже, архівація в інформаційних системах виконує не одну, а комплексну роль - організаційну, технічну, економічну й частково захисну.

3. Відмінність між архівацією, стисненням і резервним копіюванням

У практиці роботи з даними дуже часто плутають три поняття - архівацію, стиснення та резервне копіювання. Насправді вони взаємопов’язані, але не тотожні.

Стиснення - це процес зменшення обсягу подання даних завдяки використанню спеціальних алгоритмів. Його головна мета - скоротити кількість байтів, потрібних для зберігання або передавання інформації. Наприклад, якщо текстовий файл містить багато повторюваних структур, алгоритм стиснення може подати його коротше.

Архівація - це ширше поняття. Вона передбачає створення спеціального архівного контейнера, в який об’єднуються файли та папки. Архівація може включати стиснення, але не обов’язково зводиться лише до нього. Головна особливість архівації полягає в упаковці даних у впорядковану структуру, з якою зручно працювати як з єдиним об’єктом.

Резервне копіювання - це створення копії даних для подальшого відновлення у випадку втрати, пошкодження, вірусної атаки, помилки користувача чи апаратного збою. Головна мета резервного копіювання - забезпечити відновлення працездатної версії інформації. Резервна копія може бути архівованою або стиснутою, але її основне завдання - не компактність, а надійне збереження дубліката.

Розглянемо простий приклад. Якщо користувач об’єднав папку з курсовою роботою у ZIP-файл - це архівація. Якщо цей ZIP-файл став меншим за початкову папку - архівація супроводжувалася стисненням. Якщо ж цей архів збережено на зовнішньому носії для можливого відновлення в разі втрати даних - він одночасно виконує функцію резервної копії.

Таким чином, стиснення відповідає за зменшення обсягу, архівація - за упаковку та організацію, а резервне копіювання - за створення відновлюваної копії. Для грамотної роботи з інформаційними системами ці поняття необхідно чітко розрізняти.

4. Принципи роботи архіваторів

Архіватор - це програмний засіб, який призначений для створення, перегляду, зміни, розпакування та перевірки архівних файлів. Принцип його роботи ґрунтується на послідовному виконанні кількох дій.

На першому етапі архіватор аналізує вибрані користувачем файли та папки. Програма зчитує метадані - імена файлів, структуру каталогів, розмір, час створення чи зміни та інші службові характеристики. Це потрібно для того, щоб після розпакування можна було відновити початкову структуру даних.

На другому етапі архіватор формує архівний контейнер. Якщо обрано режим без стиснення, програма просто упаковує файли в єдину структуру. Якщо ввімкнено стиснення, архіватор додатково застосовує один або кілька алгоритмів, які намагаються зменшити обсяг даних. Ефективність цього процесу залежить від типу файлів. Наприклад, текстові та службові дані часто стискаються добре, а вже стиснені відео, зображення або музичні файли - значно гірше.

На третьому етапі архіватор записує службову інформацію, необхідну для подальшого відкриття архіву. У ній можуть зберігатися таблиці вмісту, відомості про структуру папок, параметри стиснення, контрольні суми, позначення шифрування й інші технічні дані.

Під час розпакування архіватор виконує зворотну процедуру. Він відкриває архів, перевіряє його цілісність, при потребі розшифровує вміст, відновлює стиснені дані до початкового вигляду і створює на носії файли та каталоги в тій структурі, яку було зафіксовано під час архівації.

Якщо архів пошкоджено, архіватор може повідомити про помилку читання, неповноту даних або неможливість повного відновлення. Саме тому якісні архіватори часто мають функції перевірки архіву, тестування вмісту й інколи навіть часткового відновлення.

Отже, архіватор - це не просто «програма для стискання», а повноцінний інструмент керування впорядкованими пакетами даних.

5. Основні формати архівів

У практиці використовують багато форматів архівів, кожен із яких має власні технічні особливості, рівень поширеності та сферу доцільного застосування.

Формат ZIP є одним із найпоширеніших у світі. Його головна перевага - універсальність і сумісність. Більшість операційних систем підтримують ZIP без встановлення додаткових програм. Саме тому цей формат часто використовують у повсякденній роботі - для надсилання документів, навчальних матеріалів, звітів, невеликих наборів файлів і загального обміну інформацією.

Формат RAR довгий час був популярним завдяки хорошому ступеню стиснення, підтримці багатотомності та додатковим можливостям, пов'язаним із захистом і частковим відновленням архівів. Його часто застосовували для великих масивів даних, програмних дистрибутивів, резервних комплектів документів.

Формат 7Z відомий високою ефективністю стиснення та широкими можливостями налаштування. Він особливо корисний тоді, коли потрібно отримати максимально компактний архів. У професійній практиці цей формат часто використовують для великих наборів текстових, технічних або змішаних даних.

Формат TAR сам по собі зазвичай не виконує стиснення, а працює як контейнер для об'єднання файлів. Найчастіше його поєднують з алгоритмами додаткового стискання, утворюючи файли типу TAR.GZ, TAR.BZ2 або TAR.XZ. Такий підхід характерний для операційних систем сімейства UNIX та Linux, де TAR використовується дуже широко.

Формати GZ, BZ2, XZ зазвичай пов'язані з потоковим стисненням окремих файлів або TAR-контейнерів. Вони особливо поширені в системному адмініструванні, у дистрибутивах програмного забезпечення та в інструментах командного рядка.

Вибір формату архіву залежить від кількох чинників - сумісності, потрібного рівня стиснення, наявності додаткових функцій, операційної системи, розміру файлів і зручності подальшого використання. Для повсякденної універсальної роботи найчастіше застосовують ZIP, для глибшого стиснення - 7Z, а для серверного та системного середовища - TAR з додатковими алгоритмами стиснення.

6. Види архіваторів та їх функції

Архіватори можна класифікувати за способом використання, призначенням і функціональними можливостями.

Першу групу становлять архіватори загального призначення. Це програми з графічним інтерфейсом, орієнтовані на широке коло користувачів. Вони дозволяють створювати архіви, розпаковувати файли, переглядати вміст, додавати або видаляти елементи, встановлювати пароль, змінювати параметри стиснення, перевіряти архіви на цілісність. Такі програми найбільш зручні у щоденній роботі.

Другу групу становлять системні архіватори, які часто працюють через командний рядок. Вони активно використовуються адміністраторами, розробниками, DevOps-фахівцями та тими, хто автоматизує обробку даних. Їх перевага полягає в тому, що вони легко інтегруються в скрипти, засоби резервного копіювання, системи розгортання й автоматизовані процедури обслуговування інфраструктури.

Третю групу утворюють вбудовані засоби операційних систем. Вони мають базову функціональність і найчастіше підтримують роботу з ZIP-архівами. Такі інструменти зручні тим, що доступні без додаткового встановлення програм, але зазвичай мають обмежені налаштування й менший набір професійних можливостей.

Четверта група - це онлайн-архіватори. Вони дозволяють стискати або розпаковувати файли через веб-інтерфейс. Такі сервіси можуть бути корисними в окремих ситуаціях, але для конфіденційних або службових даних їх використання є ризикованим, оскільки інформація передається сторонньому сервісу.

До основних функцій архіваторів належать створення архівів, розпакування, перегляд внутрішньої структури архіву, тестування вмісту, встановлення пароля, шифрування, створення багатотомних архівів, генерація саморозпаковуваних файлів, керування рівнем стиснення та інколи відновлення частково пошкоджених архівів.

Таким чином, архіватори - це не однорідна група програм, а набір інструментів, які можуть бути адаптовані як до потреб звичайного користувача, так і до складних професійних сценаріїв.

7. Багатотомні та саморозпаковувані архіви

Багатотомні архіви створюють тоді, коли загальний архівний файл занадто великий і його зручно або необхідно розділити на кілька частин. Кожна така частина називається томом. Разом вони складають єдиний архів, який можна зібрати й розпакувати за наявності всіх томів.

Потреба в багатотомності виникає в різних ситуаціях. Наприклад, якщо дані потрібно записати на кілька носіїв обмеженої місткості, передати мережею частинами, зберігати у вигляді окремих порцій або надсилати через системи, які мають обмеження на розмір одного файла. Це дає змогу зручніше працювати з великими архівними наборами.

Однак багатотомність має і певні обмеження. Якщо хоча б один том буде втрачено або пошкоджено, повне відновлення архіву може стати неможливим. Тому багатотомні архіви потребують уважного контролю комплектності.

Саморозпаковувані архіви - це архіви, в які вбудовано модуль автоматичного розпакування. Зазвичай такий архів має вигляд виконуваного файла. Користувач може відкрити його без встановлення окремого архіватора, якщо операційна система дозволяє запуск відповідних файлів.

Перевага саморозпаковуваних архівів полягає у зручності. Їх можна передавати людям, які не мають спеціального програмного забезпечення для роботи з архівами. Такі архіви часто використовують для поширення інсталяційних комплектів, технічних матеріалів або внутрішніх пакетів документів.

Разом з тим саморозпаковувані архіви несуть певний ризик. Оскільки вони є виконуваними файлами, їх можуть використовувати для маскування шкідливих програм. Саме тому в професійній практиці до таких архівів ставляться з обережністю, особливо якщо вони походять із ненадійних джерел.

8. Захист архівів

Архіви часто використовують не лише для зручного зберігання, а й як засіб базового захисту інформації. Найпростішим способом такого захисту є парольний доступ. У цьому випадку архів можна відкрити або розпакувати лише після введення правильного пароля.

Парольний захист корисний для передавання документів, які не повинні бути доступні стороннім особам. Наприклад, це можуть бути звіти, навчальні відомості, проєктні матеріали, внутрішні документи організації. Однак надійність такого захисту прямо залежить від складності пароля. Простий пароль можна підібрати, а тому формальний захист без продуманого підходу не дає реальної безпеки.

Більш надійним варіантом є шифрування архіву. У такому випадку захищається не лише можливість відкриття архіву, а й сам його вміст. Навіть якщо зловмисник отримає доступ до файлу архіву, без ключа або правильного пароля він не зможе прочитати збережені всередині дані.

Ще одним елементом захисту є контроль цілісності. Деякі архіватори використовують контрольні суми та механізми перевірки, які дозволяють визначити, чи не було архів пошкоджено або змінено. Це особливо важливо під час мережевого передавання, довготривалого зберігання або резервування.

Разом з тим потрібно розуміти, що захищений архів не є повноцінною заміною комплексної інформаційної безпеки. Якщо пароль передано ненадійним каналом або зберігається в незахищеному місці, архів стає вразливим. Отже, захист архівів має бути частиною загальної культури безпечної роботи з даними.

9. Практичне застосування архівації

Архівація використовується в багатьох сферах навчальної, професійної та системної діяльності. У навчальному процесі архіви зручні для передавання лабораторних робіт, курсових проєктів, презентацій, методичних матеріалів, великих комплектів файлів для перевірки або спільної роботи. Замість того щоб завантажувати окремо десятки файлів, студент або викладач може працювати з одним упорядкованим архівом.

У професійній офісній діяльності архівація дає змогу об’єднувати документи, договори, звіти, таблиці, презентації, технічні матеріали та робочі папки у зручні пакети. Це корисно як для передавання, так і для довготривалого зберігання проєктної документації.

У сфері розробки програмного забезпечення архіви часто використовують для передавання збірок, вихідного коду, документації, шаблонів конфігурацій, бібліотек, логів тестування та резервних копій середовища проєкту. Архівування дозволяє зберігати пов’язаний набір файлів у цілісному вигляді.

У системному адмініструванні архівація є стандартним інструментом. Вона використовується для упаковки журналів подій, резервних копій конфігурацій, знімків системного стану, дистрибутивів оновлень і комплектів для перенесення сервісів між середовищами.

У хмарних і мережевих системах архівація сприяє зменшенню трафіку, спрощує синхронізацію і прискорює передавання великих наборів файлів. Це особливо важливо тоді, коли йдеться про віддалену роботу, міжсерверний обмін або централізоване зберігання даних.

Таким чином, архівація є не лише технічною процедурою, а й важливим елементом організації цифрової діяльності.

10. Переваги та недоліки архівації

Архівація має низку очевидних переваг. Найперше вона дозволяє зібрати багато файлів у єдиний контейнер, що значно спрощує зберігання, пересилання та облік матеріалів. Другим важливим плюсом є можливість зменшення загального розміру даних, якщо вміст добре піддається стисненню. Третя перевага полягає в тому, що архіви спрощують структуризацію інформації - завершені проєкти, старі версії документів чи тематичні комплекти можна зберігати як окремі архівні пакети.

Ще одна перевага - можливість захисту даних за допомогою пароля та шифрування. Для повсякденного обміну конфіденційною інформацією це дуже корисно. Крім того, архівація полегшує резервування та перенесення матеріалів між пристроями.

Проте архівація має й недоліки. По-перше, не всі типи файлів добре стискаються. Деякі мультимедійні формати, зашифровані дані або вже стиснені архіви майже не дають виграшу в розмірі. По-друге, якщо архів пошкодиться, під загрозою може опинитися одразу весь набір файлів, що містився всередині. По-третє, забутий пароль може зробити дані недоступними навіть для законного користувача.

Також слід враховувати, що архівація потребує часу на створення та розпакування, а іноді - додаткових ресурсів процесора й пам’яті. У великих системах або при роботі з дуже масивними архівами це може бути відчутним фактором.

Отже, архівація є дуже корисним інструментом, але її слід застосовувати з урахуванням конкретної задачі, типу даних і ризиків.

11. Типові помилки під час архівації

На практиці користувачі часто роблять помилки, які знижують ефективність архівації або створюють загрозу втрати даних. Однією з найпоширеніших помилок є ототожнення архіву з резервною копією. Якщо користувач має лише один архівний файл і не зберігає його дублікати окремо, у разі пошкодження архіву можна втратити всю інформацію.

Друга типова помилка - невиконання перевірки архіву після створення. Іноді користувачі вважають, що якщо архів створився без явної помилки, то він гарантовано коректний. Насправді доцільно перевіряти, чи відкривається архів і чи можна переглянути його вміст.

Третя помилка пов’язана зі слабким захистом. Ненадійні паролі або збереження пароля в очевидному місці знижують цінність захищеного архіву майже до нуля. Четверта помилка - відкривання архівів із ненадійних джерел без перевірки безпечності. Архів може містити шкідливі файли, скрипти або замасковані виконувані модулі.

П’ята поширена помилка - беззмістовна повторна архівація вже стиснених даних. Це майже не дає виграшу, але витрачає час і ресурси. Шоста помилка - відсутність зрозумілої системи іменування архівів. Якщо архіви названі хаотично, без дат, версій або тематичних ознак, з часом знайти потрібний файл стає складно.

Для професійної роботи важливо виробити дисциплінований підхід: перевіряти архіви, дублювати важливі дані, використовувати зрозумілі назви, не покладатися на один єдиний контейнер і не відкривати підозрілі архіви без попереднього аналізу.

Архівація даних є важливою технологією організації цифрової інформації. Вона дозволяє об'єднувати файли в єдиний контейнер, зменшувати їх розмір, спрощувати передавання, упорядковувати зберігання та забезпечувати базовий рівень захисту.

У сучасних інформаційних системах архівація має практичне значення в навчальній роботі, офісному документообігу, розробці програмного забезпечення, системному адмініструванні, хмарних сервісах і резервуванні. Вона не є повною заміною резервного копіювання чи інформаційної безпеки, але є важливим інструментом ефективної роботи з даними.

Для майбутнього фахівця з інформаційних технологій розуміння архівації є обов'язковим, оскільки ця технологія широко використовується майже в усіх цифрових середовищах. Грамотне застосування архіваторів, правильний вибір формату архіву, уважне ставлення до безпеки та уникнення типових помилок дозволяють значно підвищити ефективність роботи з інформацією.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Івашко А. В. Теорія інформації та кодування в прикладах і задачах : навч. посібник / А. В. Івашко, В. А. Крилова ; Нац. техн. ун-т «Харків. політехн. ін-т». Харків : НТУ «ХПІ», 2022. 317 с. URL: електронний ресурс (дата звернення: 19.06.2026).
2. Майданюк В. П. Основи теорії інформації та кодування : навчальний посібник / В. П. Майданюк, О. Н. Романюк, С. Є. Тужанський. Вінниця : ВНТУ, 2022. 133 с. URL: електронний ресурс (дата звернення: 19.06.2026).
3. Прокопишин І. А. Основи теорії інформації та кодування : навч. посібник / І. А. Прокопишин, Р. Є. Рикалюк, В. Ф. Чекурін, К. А. Червінка. Електрон. вид. Львів : ЛНУ ім. Івана Франка, 2023. 156 с. URL: електронний ресурс (дата звернення: 19.06.2026).
4. Гнусов Ю. В. Теорія інформації та кодування : навчальний посібник / Ю. В. Гнусов, В. В. Носов. Харків : ХНУВС, 2023. 212 с.
5. Гайдур Г. І. Теорія інформації та кодування : навчальний посібник для підготовки до практичних занять / Г. І. Гайдур, З. З. Бондаренко. Київ : ДУІКТ, 2024. 43 с. URL: електронний ресурс (дата звернення: 19.06.2026).
6. Polyanskiy Y. Information Theory: From Coding to Learning / Y. Polyanskiy, Y. Wu. Cambridge : Cambridge University Press, 2025. URL: electronic resource (date of access: 19.06.2026).
7. Guruswami V. Essential Coding Theory / V. Guruswami, A. Rudra, M. Sudan. Draft version. 2026. URL: electronic resource (date of access: 19.06.2026).
8. Vinck A. J. H. Coding Concepts and Reed-Solomon Codes. 2022. URL: electronic resource (date of access: 19.06.2026).
9. Kadir W. K. Efficient Interpolation-Based Decoding of Reed-Solomon Codes / W. K. Kadir, H.-Y. Lin, E. Rosnes. 2023. URL: electronic resource (date of access: 19.06.2026).
10. Alrabiah O. Randomly punctured Reed-Solomon codes achieve list-decoding capacity over linear-sized fields / O. Alrabiah, V. Guruswami, R. Li. 2023. URL: electronic resource (date of access: 19.06.2026).
11. ISO/IEC 18004:2024. Information technology — Automatic identification and data capture techniques — QR Code bar code symbology specification. Geneva : International Organization for Standardization, 2024. URL: electronic resource (date of access: 19.06.2026).
12. GS1 General Specifications Standard. Release 25.0. GS1, 2025. URL: electronic resource (date of access: 19.06.2026).
13. Collet Y. Zstandard Compression and the “application/zstd” Media Type / Y. Collet, M. Kucherawy. RFC 8878. IETF, 2021. URL: electronic resource (date of access: 19.06.2026).
14. Pavlov I. LZMA SDK : software development kit. Version 26.01. 7-Zip, 2026. URL: electronic resource (date of access: 19.06.2026).
15. 7-Zip : file archiver with LZMA and LZMA2 compression. 2026. URL: electronic resource (date of access: 19.06.2026).
16. Linear Error-Correcting Codes : lecture notes. MIT OpenCourseWare, 2024. URL: electronic resource (date of access: 19.06.2026).
17. Blackledge J. Coding Theory — Advances and Applications in Informatics. IntechOpen, 2025. URL: electronic resource (date of access: 19.06.2026).
18. Mouloua E. M. Foundations of Information Theory for Coding Theory / E. M. Mouloua, E. Mohamed. 2025. URL: electronic resource (date of access: 19.06.2026).
19. Niu K. A Mathematical Theory of Semantic Communication / K. Niu, P. Zhang. 2024. URL: electronic resource (date of access: 19.06.2026).
20. Noever D. Dueling QR Codes: The Hyding of Dr. Jeckyl / D. Noever, F. McKee. 2025. URL: electronic resource (date of access: 19.06.2026).

Т 33 **Теорія інформації та кодування**: конспект лекцій для здобувачів першого (бакалаврського) рівня вищої освіти освітньої програми «Інформаційні системи та технології охорони і безпеки» галузі знань F (12) Інформаційні технології спеціальності F6 (126) Інформаційні системи та технології денної та заочної форм навчання / уклад. С.В. Гринюк, М.М. Поліщук. Луцьк: ЛНТУ, 2026. 88 с.

Комп’ютерний набір: С.В. Гринюк
Редактор: С.В. Гринюк

Підп. до друку _____ 2026 р.
Формат 60х84/16. Папір офс. Гарнітура Таймс.
Ум. друк. арк.____Тираж ___ прим. Зам._____

Відділ іміджу та промоції
Луцького національного технічного університету
43018 м. Луцьк, вул. Львівська, 75