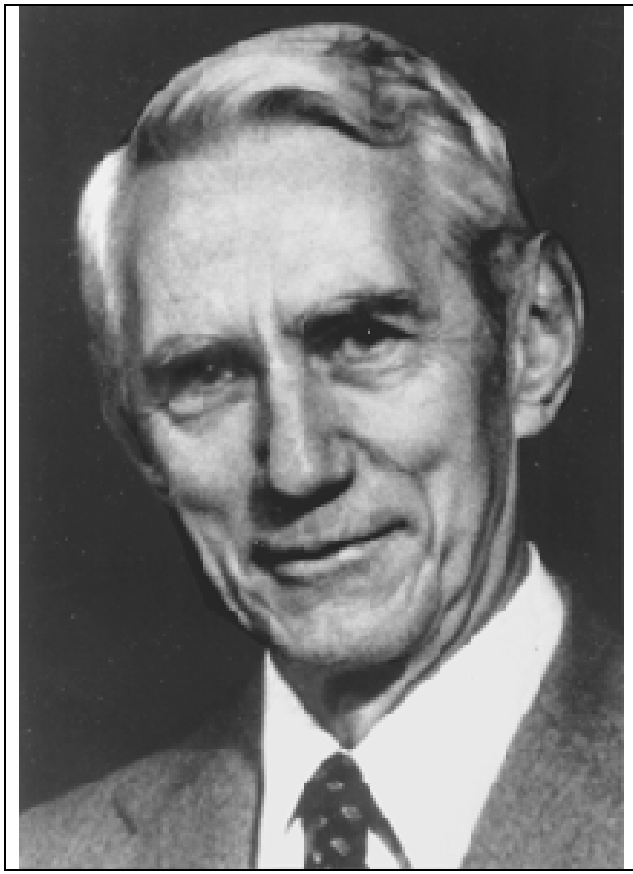
Тема 23. Алфавітне й рівномірне кодування. Роздільні коди. Оптимальне кодування

План лекції

- **Алфавітне й рівномірне кодування**
- **Достатні умови однозначності декодування. Властивості роздільних кодів.**
- **Оптимальні коди. Код Фано — код, близький до оптимального**

Кодування – у широкому розумінні – це перехід від одного способу подання інформації до іншого, який дає змогу відновлювати початкову інформацію. Теорія кодування виникла в 40-х роках XX століття після робіт К. Шеннона. У цій теорії досліджуються методи кодування, пов'язані з основними математичними моделями, які відображають істотні риси реальних інформаційних систем.



Клод Елвуд Шеннон — американський електротехнік і математик, «батько теорії інформації». Шеннон відомий тим, що запропонував теорію інформації в науковій статті, опублікованій в 1948 році.

Народився: 30 квітня 1916 р., Мічиган, США
Помер: 24 лютого 2001 р., Медфорд, Массачусетс, США
Нагороди: Медаль пошани IEEE, Національна наукова медаль США в галузі технічних наук
Освіта: Мічиганський університет, Массачусетський технологічний інститут

Теорема про пропускну здатність каналу: Будь-який канал з шумом характеризується максимальною швидкістю передачі повідомлення, ця межа названа в честь Шеннона. При передачі інформації зі швидкостями, які перевищують цю межу, відбувається невідворотне спотворення даних, але знизу до цієї межі можна наближатися з необхідною точністю, забезпечуючи задано малу ймовірність помилки передачі інформації в зашумленому каналі.

Алфавітне й рівномірне кодування

Без утрати загальності можна сформулювати задачу кодування так. Нехай задано алфавіт $A = \{a_1, \ldots, a_r\}$ зі скінченної кількості символів, які називають *буквами*. Скінченну послідовність букв $a_{i_1} a_{i_2} \ldots a_{i_l}$ алфавіту A називають *словом* у цьому алфавіті. Для позначення слів ми будемо використовувати малі грецькі букви: $\alpha = a_{i_1} a_{i_2} \ldots a_{i_l}$. Число l – кількість букв у слові α – називають *довжиною* слова α та позначають $l(\alpha)$. Множину всіх слів у алфавіті A позначають як A^*. Порожнє слово позначають λ; зазначимо, що $\lambda \notin A$, $\lambda \in A^*$, $l(\lambda)=0$.

Якщо слово α має вигляд $\alpha = \alpha_1\alpha_2$, то α_1 називають *початком* (*префіксом*) слова α, а α_2 – *закінченням* (*постфіксом*) слова α. Якщо при цьому $\alpha_1 \neq \lambda$ (відповідно $\alpha_2 \neq \lambda$), то α_1 (відповідно α_2) називають *власним початком* (відповідно *власним закінченням*) слова α.

Нехай S – підмножина множини A^*: $S \subset A^*$; елементи множини S називають *повідомленнями*. Нехай також задано скінченний алфавіт $B = \{b_1, \ldots, b_q\}$. Як β позначимо слово в алфавіті B і як B^* – множину всіх слів в алфавіті B. Задамо відображення f, яке кожному слову α, $\alpha \in S$, ставить у відповідність слово $\beta = f(\alpha)$, $\beta \in B^*$. Слово β називають *кодом повідомлення* α. Перехід від слова α до його коду називають *кодуванням*.

Якщо $|B| = q$, то кодування називають *q-ковим*. Найчастіше використовують алфавіт $B = \{0, 1\}$, тобто *двійкове кодування*. Саме його ми й розглядатимемо в наступних підрозділах, випускаючи слово «двійкове».

Відображення f задають якимось алгоритмом. Є два способи задати відображення f.

1. Алфавітне кодування. *Алфавітне* кодування задають *схемою* (або *таблицею кодів*) σ:

$$a_1 \to \beta_1,$$
$$a_2 \to \beta_2,$$
$$\ldots\ldots\ldots$$
$$a_r \to \beta_r,$$

де $a_i \in A$, $\beta_i \in B^*$, $i = 1, \ldots, r$.

Схема σ задає відповідність між буквами алфавіту A та деякими словами в алфавіті B. Вона визначає алфавітне кодування так: кожному слову $\alpha = a_{i_1} a_{i_2} \ldots a_{i_l}$ із множини S ставиться у відповідність слово $\beta = \beta_{i_1}\beta_{i_2}\ldots\beta_{i_l}$. Це слово β називають *кодом* повідомлення α. Слова $\beta_1, \ldots, \beta_r$ (див. схему σ) називають *елементарними кодами*.

Схеми алфавітного кодування використовують, зокрема, для побудови оптимальних кодів, або кодів з мінімальною надлишковістю.

2. Рівномірне кодування. *Рівномірне* (або *блокове*) *кодування* з *параметрами k та n* визначають так. Повідомлення α розбивають на блоки довжиною k:

$$\alpha=(x_1\ldots x_k)\ (x_{k+1}\ldots x_{2k})\ \ldots\ (x_{mk+1}\ldots x_{mk+s}),$$

де $s\le k$ (останній блок може бути коротшим, у такому разі спосіб його кодування спеціально обумовлюють), $x_i\in A$, $i=1,\ldots, mk+s$.

Блоки довжиною k розглядають як „букви" якогось алфавіту (таких блоків є, очевидно, r^k, бо алфавіт A складається з r букв) і кодують словами в алфавіті B довжиною n за *схемою рівномірного кодування* $\sigma_{k,n}$:

$$\alpha_1\to\beta_1,$$
$$\alpha_2\to\beta_2,$$
$$\alpha_3\to\beta_3,$$
$$\ldots\ldots\ldots$$
$$\alpha_{r^k}\to\beta_{r^k}.$$

На практиці використовують значення $r=2$, тобто двійкові блоки довжиною k кодують двійковими словами довжиною n, де $n>k$. Такі схеми рівномірного кодування використовують для побудови кодів, стійких до перешкод у каналах зв'язку. Головними формами стійкості коду до перешкод є здатність *виявляти деяку кількість помилок*, а також сильніша властивість – *виправляти помилки*. Це досягається введенням певної надлишковості.

Надлишковістю схеми $\sigma_{k,n}$ *на символ повідомлення* називають величину $R=\dfrac{n-k}{k}=\dfrac{n}{k}-1.$

Отже, ми описали дві можливості задати відображення f: $S \to B^*$. Однозначне декодування можливе, якщо існує обернене відображення f^{-1}.

Достатні умови однозначності декодування. Властивості роздільних кодів

Розглянемо схему алфавітного кодування σ та різні слова, складені з елементарних кодів. Схему σ називають *роздільною*, якщо з того що

$$\beta_{i_1} \ldots \beta_{i_k} = \beta_{j_1} \ldots \beta_{j_l}$$

випливає, що $k = l$ та $i_t = j_t$ для кожного $t = 1, \ldots, k$, тобто будь-яке слово, складене з елементарних кодів, можна єдиним способом розкласти на елементарні коди. Очевидно, що алфавітне кодування з роздільною схемою вможливлює однозначне декодування.

Схему σ називають *префіксною*, якщо для будь-яких i та j ($i, j=1, \ldots, r$; $i \neq j$) елементарний код β_i не є префіксом елементарного коду β_j.

Теорема 1. Префіксна схема є роздільною.

Доведення. Від протилежного. Оскільки схема σ префіксна, то всі елементарні коди в σ попарно різні, тобто $\beta_i \neq \beta_j$, якщо $i \neq j$. Нехай кодування зі схемою σ не роздільне. Тоді існує таке слово $\beta \in B^*$, що можуть бути два розклади на елементарні коди:

$$\beta = \beta_{i_1} \ldots \beta_{i_k}, \ \beta = \beta_{j_1} \ldots \beta_{j_l}.$$

Нехай $\beta_{i_1} = \beta_{j_1}$, …, $\beta_{i_{p-1}} = \beta_{j_{p-1}}$ але $\beta_{i_p} \neq \beta_{j_p}$. У такому разі одне зі слів β_{i_p}, β_{j_p} являє собою префікс іншого, а це суперечить тому, що схема σ префіксна.

Властивість префіксності достатня, але не необхідна умова роздільності схеми.

Приклад. Нехай $A=\{a, b\}$, $B=\{0, 1\}$, схема σ: $a\to 0$, $b\to 01$. Ця схема не префіксна, але роздільна. Справді, перед кожним входженням 1 в слові β_2 безпосередньо є 0. Це дає змогу виділити всі пари (01). Частина слова, що залишилась, складатиметься із символів 0.

Нехай $\beta = y_1 y_2 \ldots y_n$ – слово з B^*. Позначимо як $\tilde{\beta}$ слово, одержане оберненням слова β, тобто $\tilde{\beta} = y_n y_{n-1} \ldots y_1$. Позначимо як $\tilde{\sigma}$ схему

$$a_1 \to \tilde{\beta}_1,$$
$$a_2 \to \tilde{\beta}_2,$$
$$\ldots\ldots\ldots$$
$$a_r \to \tilde{\beta}_r.$$

Приклад. Візьмемо схему σ з попереднього прикладу. Тоді $\tilde{\sigma}$ має такий вигляд: $a\to 0$, $b\to 10$. Схема $\tilde{\sigma}$ префіксна, тому за теоремою 1 – роздільна.

Зауваження. Схеми σ та $\tilde{\sigma}$ водночас або роздільні, або ні.

Це зауваження дає змогу посилити теорему 1.

Теорема 2. Якщо або схема σ, або схема $\tilde{\sigma}$ префіксна, то обидві схеми σ та $\tilde{\sigma}$ роздільні.

Можна навести приклад такої роздільної схеми σ, що ні σ, ні $\tilde{\sigma}$ не префіксні (див. задачу 2 на стор. 297 підручника). Отже, теорема 2 також дає достатню, але не необхідну умову роздільності схеми.

Нехай задано схему алфавітного кодування σ: $a_i \to \beta_i$, $a_i \in A$, $\beta_i \in B^*$, $i = 1, \ldots, r$. Для того, щоб схема алфавітного кодування була роздільною, необхідно, щоб довжини елементарних кодів задовольняли *нерівність Мак-Міллана* (B. McMillan).

Теорема 3 (нерівність Мак-Міллана). Якщо схема алфавітного кодування σ роздільна, то

$$\sum_{i=1}^{r} \frac{1}{2^{l_i}} \le 1, \text{ де } l_i = l(\beta_i).$$

Зауваження 1. Зазначимо, що нерівність Мак-Міллана – це необхідна умова роздільності схеми алфавітного кодування: якщо схема роздільна, то нерівність виконується. Якщо ж нерівність виконується, то з цього НЕ випливає, що схема роздільна: є нероздільні схеми, для яких нерівність Мак-Міллана виконується, наприклад:

$$a_1 \to 1, \quad a_2 \to 01, \quad a_3 \to 10.$$

Справді, повідомлення 1101 можна розкласти на елементарні коди двома способами:
1/1/01, що дасть при декодуванні $a_1 a_1 a_2$, і 1/10/1, що дасть $a_1 a_3 a_1$.

Зауваження 2. У теоремі 3 розглянуто нерівність Мак-Міллана для двійкового кодування. Для загального випадку q-кового кодування вона має вигляд:

$$\sum_{i=1}^{r} \frac{1}{q^{l_i}} \le 1, \text{ де } l_i = l(\beta_i).$$

Розглянемо ще один важливий факт.

Теорема 4. Якщо числа $l_1, \ldots, l_r$ задовольняють нерівність

$$\sum_{i=1}^{r} \frac{1}{2^{l_i}} \le 1 \text{ (нерівність Мак-Міллана),}$$

то існує префіксна схема алфавітного кодування σ:

$$a_1 \to \beta_1,$$
$$\ldots\ldots\ldots$$
$$a_r \to \beta_r,$$

де $l_1 = l(\beta_1), \ldots, l_r = l(\beta_r)$.

Наслідок 1. Нерівність Мак-Міллана – необхідна й достатня умова існування алфавітного кодування з префіксною схемою та довжинами елементарних кодів, що дорівнюють $l_1,\ldots, l_r$.

Наслідок 2. Якщо існує роздільна схема алфавітного кодування із заданими довжинами елементарних кодів, то існує й префіксна схема з тими самими довжинами елементарних кодів.

Для доведення потрібно спочатку застосувати теорему 3, а потім – теорему 4.

Оптимальні коди. Код Фано – код, близький до оптимального

Досі термін «код» було використано в загальноприйнятому розумінні (як код повідомлення). Однак у теорії кодування, а ще раніше в техніці слово «*код*» трактують також і як *множину елементарних кодів*. Починаючи з цього місця, будемо використовувати слово «код» у двох значеннях. З контексту буде зрозуміло, яке саме з них ми маємо на увазі.

Нехай задано алфавіт $A = \{a_1, \ldots, a_r\}$ і ймовірності $P = (p_1, \ldots, p_r)$ появи букв у повідомленні; тут p_i – ймовірність появи букви a_i. Не зупиняючись на строгому означенні ймовірності, зауважимо лише, що ймовірність деякої букви можна уявити як частину випадків, у яких вона з'являється, від загальної кількості букв у повідомленні. Так, якщо $p_1 = 0.25$, то це означає, що з, наприклад, 1000 переданих букв близько 250 разів з'явитися буква a_1. Не втрачаючи загальності, можна вважати, що

$$p_1 \geq p_2 \geq \ldots \geq p_r > 0,$$

тобто можна одразу вилучити букви, які не можуть з'явитись у повідомленні, і впорядкувати букви за спаданням ймовірностей їх появи. Крім того, $p_1 + p_2 + \ldots + p_r = 1$. Кортеж $P = (p_1, \ldots, p_r)$ називають *розподілом ймовірностей* (появи букв алфавіту A); це певна характеристика алфавіту повідомлень.

Для кожної роздільної схеми алфавітного кодування σ величину

$$l^{\sigma}_{сер}(P) = \sum_{i=1}^{r} p_i l_i ,$$

де $l_i = l(\beta_i)$, $i = 1, \ldots, r$, і називають *середньою довжиною* кодування за схемою σ для розподілу ймовірностей P.

Можна побачити, що величина $l^{\sigma}_{сер}$ $\left(l^{\sigma}_{сер} > 1\right)$ показує, у скільки разів зросте середня довжина слова в разі кодування зі схемою σ.

Можна довести, що мінімальне значення

$$l_* = \min_{\sigma} l_{сер}^{\sigma}$$

обов'язково досягається на якійсь схемі алфавітного кодування σ.

Коди, визначені схемою σ з $l_{сер} = l_*$, називають *кодами з мінімальною надлишковістю* (або *оптимальними кодами*) для розподілу ймовірностей *P*. За наслідком 2 з теорем 3 та 4 існує алфавітне кодування з префіксною схемою, яке дає оптимальні коди. У зв'язку з цим, будуючи оптимальні коди, можна обмежитися лише префіксними схемами.

Проста ідея побудови коду, близького до оптимального, належить американському вченому в галузі інформатики Р. Фано (R. Fano). Сформулюємо алгоритм кодування за методом Фано.

1. Упорядковуємо букви алфавіту *A* за спаданням імовірностей їх появи в повідомленні.

2. Розбиваємо множину букв, записаних у зазначеному порядку, на дві послідовні (тобто без перестановок букв) частини так, щоб сумарні ймовірності кожної з них були якомога близькими одна до одної. Кожній букві з першої частини приписуємо символ 0, другої – символ 1. Далі те саме робимо із кожною частиною, якщо вона містить принаймні дві букви. Процедуру продовжуємо доти, доки всю множину не буде розбито на окремі букви.

Роберт Фано
11 листопада 1917 р. (м. Турин, Італія – 13 липня 2016 р.(м. Нейплз, Флорида, США)
Освіта: Массачусетський технологічний інститут
У 1939 р переїхав у США. У 1961 р. під керівництвом його та Фернандо Корбато було створено систему, яка підтримувала режим розподілу часу (Compatible Time-Sharing System, CTSS). Автор коду Фано. У 1963 р. очолив проект MAC і залишався на цій посаді до 1968 року. Фото 30.05.2014 зі святкування 50 річниці проекту MAC.

Приклад. Нехай задано розподіл імовірностей P=(0.4, 0.15, 0.15, 0.15, 0.15). Побудуємо код за методом Фано. Розв'язок подано в таблиці 1.
Середня довжина коду дорівнює

$$l_{\text{сер}}^{F} = 0.4\times 2 + 0.15\times 2 + 0.15\times 2 + 0.15\times 3 + 0.15\times 3 = 2.3,$$

тут верхній індекс F означає, що код отримано методом Фано.

Таблиця 1

Букви алфавіту A	Імовірності появи букв	Розбиття множини букв			Елементарні коди
a_1	0.4	0	0		00
a_2	0.15		1		01
a_3	0.15	1	0		10
a_4	0.15		1	0	110
a_5	0.15			1	111

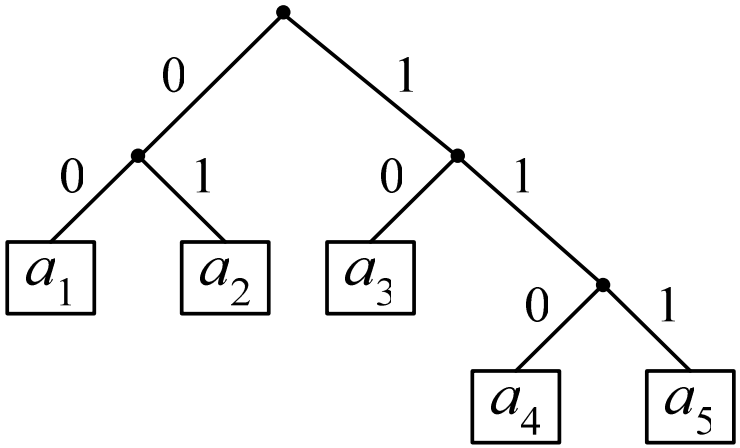


Рис. 1

Алгоритм Фано має просту інтерпретацію за допомогою бінарного дерева. Від кореня відходять два ребра, ліве позначено символом 0, а праве – символом 1. Ці два ребра відповідають розбиттю множини всіх букв на дві майже рівноймовірні частини, одній з яких поставлено у відповідність символ 0, а другій – символ 1. Ребра, що виходять із вершин наступного рівня, відповідають розбиттю одержаних частин знову на дві майже рівноймовірні послідовні частини. Цей процес продовжують доти, доки множину букв не буде розбито на окремі букви. Кожний листок дерева відповідає певному елементарному коду. Щоб виписати цей код, потрібно пройти шлях від кореня до відповідного листка.

На рисунку 1. зображено кодове дерево, отримане методом Фано для розподілу ймовірностей із останнього прикладу.

Зазначимо, що незалежно від способу кодування кожному бінарному дереву відповідає набір двійкових елементарних кодів. У такому разі дерево називають *кодовим*. Якщо елементарні коди відповідають листкам кодового дерева, то відповідна схема алфавітного кодування є префіксною (отже, забезпечується однозначність декодування).