Дослідження завадостійкого та монолітного кодування на основі комбінаторних методів

К.т.н., доц. О. Різник, к.т.н., доц. Н. Кустра, асист. Д. Скрибайло-Леськів

Lviv Polytechnic National University, 22/806 Bandera St., Lviv-13, 79013, Ukraine
E-mail: riznyk@meta.ua

Abstract. The system sectional Hamming code is considered in the real article, Cyclic Redundancy Code (CRC) and monolithic code. First from them now there is not deployment, however he shows essence of error-correcting code, while CRC is popular due to efficiency of exposure and correction of errors. In turn, a monolithic code due to a sectional structure has high efficiency.

Key words: idea ring bundle, cyclic redundancy code, Hamming code, monolithic code.

Вступ

Одним із основних способів боротьби з завадами в системах передачі та обробки інформації є застосування завадостійких кодів, що виявляють і виправляють помилки. В основі побудови завадостійких кодів лежить принцип надлишковості, що дає змогу виявляти та виправляти помилки за рахунок накладання додаткових вимог до переданих сигналів з наступною їх перевіркою.

Постановка проблеми

В цій статті розглянуто системний блочний код Хеммінга, Cyclic Redundancy Code (CRC) та монолітний код. Перший з них зараз немає широкого використання, проте він показує саму суть завадостійкого кодування, тоді як CRC є популярним за рахунок ефективності виявлення та виправлення помилок. В свою чергу, монолітний код завдяки блочній структурі має високу ефективність.

Коди Хеммінга – найбільш відомі і, ймовірно, перші з самоконтролюючих і самокоректуючих кодів. Код Хеммінга складається з двох частин. Перша частина кодує вихідне повідомлення, вставляючи в нього в певних місцях контрольні біти. Друга частина отримує вхідне повідомлення і заново обчислює контрольні біти. Якщо всі знову обчислені контрольні біти збігаються з отриманими, то повідомлення отримано без помилок. В іншому випадку, виводиться повідомлення про помилку і при можливості помилка виправляється.

Припустимо, у нас є повідомлення «html», яке необхідно передати без помилок. Для цього спочатку потрібно наше повідомлення закодувати за допомогою коду Хеммінга. Нам необхідно представити його в бінарному вигляді:

Таблиця 1

Кодове представлення символів

Символ	ASCII код	Бінарне представлення
h	68	01101000
t	74	01110100
m	6D	01101101
l	6C	01101100

На цьому етапі варто визначитися з, так званої, довжиною інформаційного слова, тобто довжиною рядки з нулів та одиниць, які ми будемо кодувати. Припустимо, у нас довжина слова буде дорівнює 16. Таким чином, нам необхідно розділити наше вихідне повідомлення («html») на блоки по 16 біт, які ми будемо потім кодувати окремо один від одного. Тобто:

$$\textit{перше\_повідомлення} = 0110100001110100$$

$$\textit{друге\_повідомлення} = 0110110101101100$$

Після цього процес кодування розпаралелюється, і дві частини повідомлення («ht» та «ml») кодуються незалежно один від одного. Розглянемо, як це робиться на прикладі першої частини. Перш за все, необхідно вставити контрольні біти. Вони вставляються в строго визначених місцях - це позиції з номерами, рівними ступеням двійки. У нашому випадку (при довжині інформаційного слова в 16 біт), це будуть позиції 1, 2, 4, 8, 16. Відповідно, у нас вийшло 5 контрольних бітів.

Тепер необхідно обчислити значення кожного контрольного біта. Значення кожного контрольного біта залежить від значень інформаційних біт, але не від усіх, а лише від тих, які цей контрольних біт контролює. Контрольний біт з номером N контролює всі наступні N біт через кожні N біт, починаючи з позиції N (рис.1).

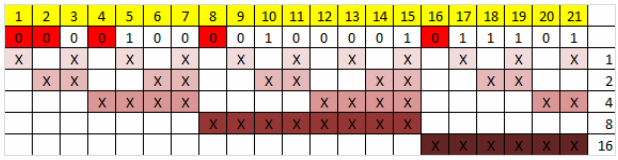

Рис.1. Обчислення контрольних бітів

Далі беремо кожен контрольний біт і дивимося скільки серед контрольованих ним бітів одиниць, отримуємо деяке ціле число і, якщо воно парне, то ставимо нуль, в іншому випадку ставимо одиницю. Можна звичайно і навпаки, якщо число парне, то ставимо одиницю, в іншому випадку, ставимо 0. Головне, щоб в «кодуючої» і «декодуючої» частинах алгоритм був однаковий. Шляхом нескладних математичних маніпуляцій ми отримуємо два закодовані повідомлення довжиною вже 21 символ (16+5):

$$\textit{перше\_закод\_повідомл} = 100011011000011010100$$

$$\textit{друге\_закод\_повідомл} = 000111011101011001100$$

Тепер, припустимо, ми отримали закодоване першою частиною алгоритму повідомлення, але воно прийшло до нас з помилкою. Наприклад ми отримали таке (11-ий біт передався неправильно):

$$\textit{передане\_повідомл} = 100011011010011010100$$

Вся друга частина алгоритму полягає в тому, що необхідно заново обчислити всі контрольні біти (так само як і в першій частині) і порівняти їх з контрольними бітами, які ми отримали. Так, порахувавши кон-

трольні біти з неправильним одинадцятий бітом ми отримаємо таку картину, де контрольні біти під номерами: 1, 2, 8 не збігаються з такими ж контрольними бітами, які ми отримали. Тепер просто склавши номера позицій неправильних контрольних біт (1+2+8=11) ми отримуємо позицію помилкового біта. Далі інвертуємо його і відкинувши контрольні біти, ми отримаємо вихідне повідомлення в первозданному вигляді. Аналогічно робимо з другою частиною повідомлення.

Циклічні коди є різновидністю поліноміальних кодів. В таких кодах вважається, що елементи $a_1, a_2, ..., a_{n-1}$ деякого кодового слова є коефіцієнтами полінома від x:

$$f(x) = a_0 + a_1 x + a_2 x^2 + ... + a_{n-1} x^{n-1} \quad (1)$$

Використовуючи це позначення можна визначити поліноміальний код як множину всіх многочленів степені не більше чим $n-1$, які містять як множник деякий фіксований многочлен $g(x)$, який називається породжувальним многочленом. Тоді процес кодування можна подати як результат множення полінома $m(x)$, що являє собою інформаційну послідовність на породжувальний многочлен $g(x)$, а декодування як результат ділення на цей поліном.

Ідея побудови циклічних кодів ґрунтується на використанні многочленів, що не приводяться (простий многочлен). Простим многочленом називається многочлен, який не може бути поданий у вигляді добутку многочленів нижчої степені, тобто такий многочлен ділиться тільки на самого себе або на 1.

Вирішення поставленої задачі

На сьогоднішній день актуальними є створення систем кодування великої потужності з високою коректувальною здатністю, для дослідження яких доцільно використати унікальні комбінаторні властивості "ідеальних кільцевих в'язанок (ІКВ), що є різновидами досконалих різницевих циклічних множин [1, 2].

ІКВ-циклічний код дозволяє виявляти до 50% і виправляти до 25% позицій кодових комбінацій, а його потужність з доповненням кодових комбінацій інверсним кодом та комбінаціями, побудованими на ізоморфних варіантах ІКВ й наближених до них комбінаторних конструкціях, може бути збільшеною в два й більше разів без значного погіршення решти їхніх показників [2, 3].

Монолітний код – це код дозволені комбінації якого утворюють послідовності з розміщених підряд однойменних символів. Суттєвою перевагою монолітного коду є простота виявлення помилок на приймальній стороні. Поява хоча б одного символу "1" серед нулів, або символу "0" серед одиниць вказує на помилку. Виключенням є поява помилки на межі між блоками одиниць і нулів. Висока завадостійкість коду забезпечується виявленням хибних символів в монолітному коді в разі їх появи. Монолітний код базується на властивостях в'язанок генерувати різні числові суми на послідовно розташованих елементах заданої послідовності. Різним позиціям монолітного коду може відповідати будь-яка закономірність розподілу ваг розрядів.

Інтерес становить реалізація кільцевого монолітного коду на ІКВ. ІКВ – це алгебраїчна структура, утворена на послідовності цілих додатних чисел, значення яких, як і значення сум поруч розміщених між собою чисел, вичерпують натуральний ряд.

Для побудови ІКВ можна використати класичні методи, що базуються на положеннях комбінаторного аналізу. Один із методів швидкої побудови ІКВ базується на понятті множника циклічної різницевої множини [1, 4, 5].

При заданій кількості n розрядів монолітного коду система кодування дає змогу кодувати будь-які числа від 1 до співвідношення $S_n = n(n-1)$.

Завадостійкість кільцевого монолітного коду (КМК) на основі ІКВ можна оцінити за співвідношенням кількості помилкових кодових комбінацій, які виявляються $M_{виявл.}$ (або виправляються $M_{випр.}$), до загальної кількості усіх можливих комбінацій заданої розрядності.

Ефективність КМК щодо можливості виявлення і виправлення помилок можна оцінити за допомогою наступних залежностей:

$$M^{(k)}_{виявл.} = \frac{C^r_{n-4}}{C^r_n} \cdot 100\%, r < n-4$$
$$M^{(k)}_{випр.} = \frac{C^r_{n-8}}{C^r_n} \cdot 100\%, r < n-8 \quad (2)$$

де n - розрядність кодового слова, а r - число помилкових символів у кодовому слові.

Висновок

Розглянутий алгоритм кодування, заснований на ІКВ, забезпечує захист даних від спотворень. В принципі будь-яка ІКВ може стати основою для синтезу завадостійкого коду. Однак слід відзначити, що за допомогою ІКВ можна будувати найбільш ефективні завадостійкі коди.

Монолітний код має низку переваг порівняно зі стандартним двійковим кодом, зокрема вищу надійність та швидкодію. Кількість слів монолітного коду з однією помилкою, яка виявляється за принципом ланцюжкової зв'язності, залежить від взаємного розміщення різнойменних символів у кодовому слові.

[1]. Різник В.В. Синтез оптимальних комбінаторних систем. - Львів, 1989.

[2]. Різник О.Я., Балич Б.І. Використання числових лінійок-в'язанок для кодування інформації. Інститутський вісник "Комп'ютерні науки та інформаційні технології", 2006. с.62-64.

[3]. Різник В.В., Скрибайло-Леськів Д.Ю. Дослідження завадостійких циклічних кодів великої потужності // Матеріали Міжнародної науково-технічної конференції SAIT 2011, с. 145.

[4]. Dimitromanolakis, A., Analysis of the Golomb Ruler and the Sidon Set Problems, and Determination of large, near−optimal Golomb Rulers, Department of Electronic and Computer Engineering Technical University of Crete, June, 2002.

[5]. http://en.wikipedia.org/wiki/Golomb_ruler.