­­МІНІСТЕРСТВО ОСВІТИ ТА НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ “ЛЬВІВСЬКА ПОЛІТЕХНІКА”

­­­

Лабораторна робота №5

з курсу «**Проблемно-орієнтовані методи та засоби**

**інформаційних тенологій**»

для студентів базового напрямку 6.08.04 "Комп’ютерні науки"

(заочна форма навчання)

Варіант 10

Виконав студент гр. КНз-3

Чалий Михайло

­­

Львів 2015

## Мета роботи

Вивчення методiв побудови кодiв Хемiнга та підходів до їх програмної реалiзацiї для практичного використання.

## Теоретичні відомості

Код Хемiнга належить до найважливiших лiнiйних кодiв, якi широко використовуються на практицi i мають зручний для технiчноi реалiзацii алгоритм виявлення та виправлення однiєї помилки.

Код складаеться з k iнформацiйних двійкових розрядів та n-k контрольних. Загальна кількість розрядів у кодi n. Нумерація розрядів здійснюється справа наліво від 1 до n.

Число n повинно задoвiльняти нерiвнiсть 2n-k не менше n+1. Для визначення основних параметрiв коду Хемiнга задається кiлькiсть iнформацiйних розрядів k, по яких обчислюється n та n-k. Кiлькiсть iнформацiйних k та контрольних n-k розрядів коду, який виявляє та коректує одну помилку приведена в таблиці.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| k | 1…4 | 5…11 | 12…26 | 27…57 | 58…120 | 121…247 |
| n-k | 3 | 4 | 5 | 6 | 7 | 8 |
| n | 7 | 15 | 31 | 63 | 127 | 255 |

На основi основних параметрiв коректуючого коду визначають позицii інформаційних та контрольних розрядів. Позиції контрольних розрядів вибираються по значенню степенi 2і, де i=0,1,2,3... Тобто номери контрольних розрядів будуть рiвнi 1,2,4,16... Між контрольними розрядами справа наліво вписуються інформаційні. Потiм визначають значення контрольних розрядів по такому правилу: сума одиниць на перевiрочних позицiях для даного контрольного розряду повинна бути непарною. Якщо ця сума непарна, то контрольний розряд встановлюється рiвним нулю, якщо парна – одиницi (доповнення до непарності).

Перевiрочнi позицii вибирають таким чином. Cкладають таблицю для ряду натуральних чисел у двiйковому кодi. Кiлькiсть чисел - n. Потiм визначають перевiрочнi позицii по такому правилу: в першу перевiрку входять розряди, якi мiстять одиницю в першому розрядi справа (1,3,5,7 і т.д.), в другу - в другому (2,3,6,7…) і т.д.

7 6 5 4 3 2 1 0

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 0 | 1 | 1 | 0 | 1 | 1 | 0 |

12 11 10 9 8 7 6 5 4 3 2 1

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 |

1100 1011 1010 1001 1000 0111 0110 0101 0100 0011 0010 0001

Рис.1. Формування коду Хемінга.

Для виявлення і виправлення одиночноi помилки в прийнятій кодовій комбінації проводять перевiрку на непарнiсть. Для першоi перевiрки сумують розряди для позицiй, що мiстять одиницю в молодшому розрядi (1,3,5,7...). Якщо кількість одиниць парна, то перший розряд справа для двійкового значення номера помилкової позиції (синдрома) встановлюється рівним одиниці. Потім сумуючи розряди для позицій, що містять одиницю у другому двійковому розряді, і перевіряючи суму на непарність визначають значення другого розряду синдрома. Процес повторюється для позицiй, що містять одиниці у розрядах 3,4,5 i т.д. В результатi формується номер помилкової позицiї коду. Для виправлення помилки символ у цiй позицii необхiдно замiнити на протилежний. Якщо синдром рівний нулю, то прийнята комбінація не містить одиночних помилок. Наприклад, при помилці в шостому розряді визначене значення синдрому буде рівне 0110.

## Завдання

* + - 1. Ознайомтесь з методом побудови коду Хемiнга.
      2. Для заданого варіанту вручну побудуйте код Хемінга, внесіть і виявіть помилку.

3. Розробiть алгоритм та реалiзуйте програму побудови коду Хемiнга.

1. Реалiзуйте програму перевiрки та виправлення кодових послiдовностей.
2. Перевiрте працездатнiсть програм на тестових прикладах.

|  |  |
| --- | --- |
| Варіант 10 | 10111101 |

## Реалізація

Лістінг 1. hamming.js

/\* jshint esnext:true \*/

function isParityBit(x){

// 1,2,4,8,16

return (x & (x - 1)) === 0;

}

function calcParity(p, input){

var pos = Math.pow(2, p-1);

var encodedPos = pos-1;

var parity1 = 0;

var parityBit = true;

var check = true;

var checkCount = 0;

var checkA = [];

for(var i = pos; i <= input.length; i++){

if (!parityBit && check){

if (input[i-1] === 1){

parity1++;

checkA.push(input[i-1]);

}

}

parityBit = false;

checkCount++;

if (checkCount >= pos){

check = !check;

checkCount = 0;

}

}

return parity1%2;

}

function encode(input){

// NOTE Most stupid implemenation possible

// Encode space

var encoded = [];

var queue = input.slice();

var parityBitsCount = 0;

while(queue.length > 0){

if (isParityBit(encoded.length + 1)){

parityBitsCount++;

encoded.push(0);

} else {

encoded.push(queue.shift());

}

}

// Set parity bits

for(var p = 1; p <= parityBitsCount; p++){

var pos = Math.pow(2, p-1);

encoded[pos-1] = calcParity(p, encoded);

}

return encoded;

}

function decode(input){

var decoded = [];

var errorParities = [];

var p = 1;

for(var i = 0; i < input.length; i++){

if (isParityBit(i+1)){

if (input[i] != calcParity(p, input)){

errorParities.push(p);

}

p++;

} else {

decoded.push(input[i]);

}

}

return {

result: decoded,

errorParities: errorParities

};

}

module.exports.isParityBit = isParityBit;

module.exports.encode = encode;

module.exports.decode = decode;

Лістінг 2. L5.js

var hamming = require('./lib/hamming');

var input = [1,0,1,1,1,1,0,1];

console.log('Вхідний код: ', input);

var encoded = hamming.encode(input);

console.log('Hamming Закодоване:', encoded);

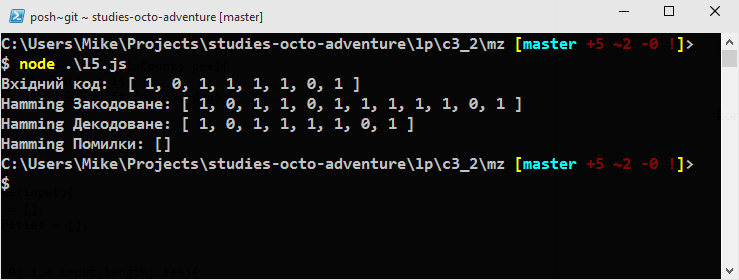
var decoded = hamming.decode(encoded);

console.log('Hamming Декодоване:', decoded.result);

console.log('Hamming Помилки:', decoded.errorParities);

## https://github.com/chaliy/studies-octo-adventure/tree/master/lp/c3\_2/mz

## Результат



## Висновки

Вивчив методи побудови кодiв Хемiнга та підходів до їх програмної реалiзацiї для практичного використання.