13. Код Грея

Код Грея — одна із систем кодування інформації, в якій два послідовні коди відрізняються значенням лише одного біта.

Для кодування ознак можна скористатись найпростішим варіантом: двійковим значенням цієї ознаки. Тоді легко використовувати бітові рядки фіксованої довжини для подання всіх можливих значень цієї ознаки.

Наприклад, десяткові числа 7 і 8 можна легко закодувати у двійкові числа B(7)=011 і B(8)=100, використовуючи двійкову техніку. Проте, якщо ми хочемо переміститися з фенотипу 7 у фенотип 8, то повинні змінити всі три біти в їх зображенні від B(7)=011 до B(8)=100. Інакше кажучи, при роботі ГА необхідні три окремі дії для переміщення від розв'язку 7 до розв'язку 8, які призведуть до додаткових витрат часу. З іншого боку, якщо ми хочемо переміститися від розв'язку 7 до розв'язку 8, то нам необхідна лише одна операція. Це ускладнює використання ГА й погіршує його збіжність.

Щоб уникнути цього, краще використовувати кодування, в якому значення розрізняються на один біт. Таким є код Грея. Розглянемо принцип його побудови:

ДесятковийКодГрея

00000

1 0001

нульовий розряд вичерпав свої ресурси (0,1) ставиться «дзеркало» і від нього «відображаються» значення 0-го розряду, але з одиницею в старшому розряді.

20011

30010

Так само і з рештою розрядів.

40110

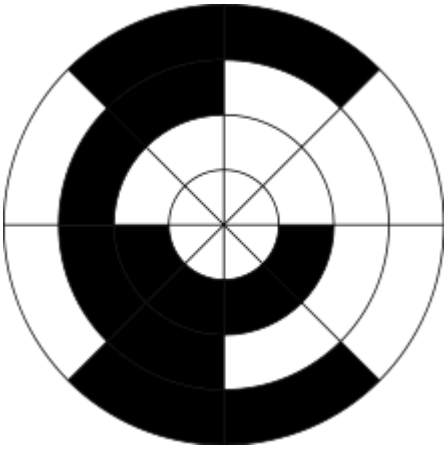
50111

60101

7 0100

81100

9 1101



Використання кодів Грея засновано насамперед на тому, що він мінімізує ефект помилок при перетворенні аналогових сигналів у цифрові (наприклад, у багатьох видах датчиків).

Коди Грея часто застосовуються в датчиках-енкодерах. Їх використання зручно тим, що два сусідніх значення шкали сигналу відрізняються лише в одному розряді. Також вони використовуються для кодування номера доріжок в твердих дисках.

Код Грея можна використовувати також і для вирішення задачі про Ханойські вежі. Широко застосовуються коди Грея і в теорії генетичних алгоритмів для кодування генетичних ознак, які представлені цілими числами. Код Грея використовується для генерації сполучень методом обертових дверей.

Коди Грея легко виходять з двійкових чисел шляхом побітової операції «виключне АБО» з тим же числом, зрушеним вправо на один біт. Отже, і-й біт коду Грея G_i виражається через біти двійкового коду B_i наступним чином:

$$G_i = B_i \oplus B_{i+1},$$

операція «виключне АБО»; біти нумеруються справа наліво, починаючи з молодшого. Нижче наведено алгоритм перетворення з двійкової системи числення в код Грея, записаний на мові C (мова програмування):

```
unsigned int grayencode(unsigned int g)
{
   return g ^ (g >> 1);
}
```

Той же алгоритм, записаний на мові Pascal:

```
function BinToGray(b:integer):integer;
begin
  BinToGray:=b xor (b shr 1)
end;
```

Приклад: перетворити двійкове число 10110 в код Грея.

```
10110
01011
-----
11101
```

Зворотний алгоритм — перетворення коду Грея в двійковий код — можна виразити рекурентною формулою

$$B_i = B_{i+1} \oplus G_i,$$

причому перетворення здійснюється побітно, починаючи зі старших розрядів, і значення Bi + 1, використовуване у формулі, обчислюється на попередньому кроці алгоритму. Дійсно, якщо підставити в цю формулу вищенаведений вираз для і-го біта коду Грея, отримаємо

$$B_i = B_{i+1} \oplus G_i = B_{i+1} \oplus (B_i \oplus B_{i+1}) = B_i \oplus (B_{i+1} \oplus B_{i+1}) = B_i \oplus 0 = B_i.$$

Однак наведений алгоритм, пов'язаний з маніпуляцією окремими бітами, незручний для програмної реалізації, тому на практиці використовують видозмінений алгоритм:

$$B_k = \bigoplus_{i=k}^{N} G_i,$$

де N — кількість бітів в коді Грея (для збільшення швидкодії алгоритму за N можна взяти

номер старшого ненульового біта коду Грея); знак означає підсумовування за допомогою операції «виключне АБО», тобто

$$\bigoplus_{i=k}^{N} G_i = G_k \oplus G_{k+1} \oplus \ldots \oplus G_{N-1} \oplus G_N.$$

Дійсно, підставивши в формулу вираз для і-го біта коду Грея, отримаємо

$$B_k = \bigoplus_{i=k}^{N} G_i = \bigoplus_{i=k}^{N} (B_i \oplus B_{i+1}) = (B_k \oplus B_{k+1}) \oplus (B_{k+1} \oplus B_{k+2}) \oplus \ldots \oplus (B_{N-1} \oplus B_N) \oplus (B_N \oplus B_{N+1}) =$$
$$= B_k \oplus (B_{k+1} \oplus B_{k+1}) \oplus \ldots \oplus (B_N \oplus B_N) \oplus B_{N+1} = B_k \oplus B_{N+1} = B_k$$

Тут передбачається, що біт, що виходить за рамки розрядної сітки ( ), дорівнює нулю. Нижче приведена функція на мові С, що реалізує даний алгоритм. Вона здійснює послідовний зсув вправо і підсумовування вихідного двійкового числа, до тих пір, поки черговий зсув не обнулить доданок.

```
unsigned int graydecode(unsigned int gray)
{
   unsigned int bin;
   for (bin = 0; gray; gray >>= 1) {
     bin ^= gray;
   }
   return bin;
}
```

Той же самий алгоритм, записаний на мові Паскаль:

```
function GrayToBin(b:integer):integer;
 var g:integer;
begin
  g:=0;
  while b>0 do begin
    g:=g xor b;
    b:=b shr 1;
  end;
  GrayToBin:=g;
end;
```

Приклад: перетворити код Грея 11101 в двійковий код.

```
11101
01110
00111
00011
00001
-----
10110
```

Швидке перетворення 8/16/24/32-разрядного значення коду Грея в двійковий код на мові BlitzBasic:

```
Function GRAY_2_BIN%(X%)
Return X Xor ((X And $88888888) Shr 4) Xor ((X And $CCCCCCCC) Shr 2) Xor ((X And $EEEEEEEE) Shr 1)
End Function
```