Koder i Dekoder Kodów BCH w Pythonie

Parametry kodu BCH

Wzory:

1. **Długość kodu**:

$n=2^m-1$

gdzie m to stopień ciała skończonego $GF(2^m)$

2. **Liczba bitów informacji**:

$k=n-\deg(g(x))$

gdzie $\deg(g(x))$to stopień wielomianu generatora.

3. **Zdolność korekcji błędów**:

$t=\lfloor\frac{d-1}{2}\rfloor$,

gdzie *d* to minimalna odległość Hamminga.

Opis:

Te wzory definiują podstawowe parametry kodu BCH: długość kodu *n*, liczbę bitów informacji *k*, oraz maksymalną liczbę błędów *t*, które kod może poprawić.

2. Generator kodu

Wzór:

Generator kodu BCH to wielomian g(x)$g(x)$, będący iloczynem minimalnych wielomianów pierwiastków ciała skończonego:

$g(x)=LCM(M_1(x),M_2(x),\ldots,M_{2t}(x))$

$g(x)$: To wielomian generatora kodu BCH. Jest to kluczowy element kodu, który definiuje sposób tworzenia zakodowanego ciągu na podstawie wiadomości wejściowej.

LCMLCM: Skrót od **least common multiple** – najmniejszy wspólny wielokrotność. Oznacza, że g(x)$g(x)$ to najmniejszy wielomian (w sensie algebraicznym), który jest podzielny przez wszystkie wymienione wielomiany $M_i(x)$.

M1(x),M2(x),…,M2t(x) - To **minimalne wielomiany** pierwiastków αiαi w ciele skończonym GF(2m). Minimalny wielomian pierwiastka to najmniejszy (w sensie stopnia) wielomian nad *GF*(2), który ma ten pierwiastek jako rozwiązanie.

Opis:

Wielomian generatora g(x) jest kluczowy w procesie kodowania. Jest on tak skonstruowany, aby umożliwić wykrycie i korekcję błędów.

Generator $g(x)$ jest wyznaczany w bibliotece na podstawie parametrów *n,k,t*, które określają długość kodu, długość wiadomości i zdolność korekcji błędów.

$$G = \begin{bmatrix} g_0 & g_1 & \cdots & g_m & 0 & \cdots & 0 \\ 0 & g_0 & g_1 & \cdots & g_m & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \ddots & 0 \\ 0 & 0 & \cdots & g_0 & g_1 & \cdots & g_m \end{bmatrix}$$

- G to macierz generatora kodu.
- $g(x)=g_0+g_1x+g_2x^2+\cdots+g_mx^m$– wielomian generatora.

Opis:

- Macierz generatora GG służy do kodowania wiadomości.
- Każdy wiersz to przesunięcie wielomianu generatora.

3. Kodowanie

Wzór:

Aby zakodować wiadomość $u(x)$, oblicza się wielomian kodowy $c(x)$:

$c(x)=u(x)\cdot g(x)$,

gdzie $u(x)$ to wielomian wiadomości (reprezentujący k-bitowy ciąg).

Opis:

Proces kodowania polega na pomnożeniu wielomianu wiadomości $u(x)$ przez generator $g(x)$. Wynikowy wielomian $c(x)$ jest gotowy do przesłania.

Program koduje wiadomość $u(x)$, mnożąc ją przez generator $g(x)$, aby utworzyć słowo kodowe $c(x)$.

Macierz Kontroli Parzystości *H*

Wzór:

$H \cdot c^T = 0$

gdzie:

- H to macierz kontroli parzystości.
- cT to transponowany wektor kodowy.

Konstrukcja macierzy H*H*:

- Kolumny H są kolejnymi wektorami nad $GF(2)$, które są liniowo niezależne.
- Typowy format H:

$$H = \begin{bmatrix} 1 & \alpha & \alpha^2 & \cdots & \alpha^{n-1} \\ 1 & \alpha^2 & (\alpha^2)^2 & \cdots & (\alpha^2)^{n-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \alpha^{2t} & (\alpha^{2t})^2 & \cdots & (\alpha^{2t})^{n-1} \end{bmatrix}$$

α – pierwiastek pierwotny ciała GF(2^m).

4. Obliczanie syndromów (dekodowanie)

Wzory:

1. Syndromy S_i oblicza się jako wartości wielomianu $r(x)$ (otrzymanego ciągu z błędami) w pierwiastkach α^i:

   $S_i=r(\alpha^i)$

   $i=1,2,\ldots,2t$

Opis:

Syndromy są używane do wykrycia, czy wystąpiły błędy. Jeśli wszystkie $S_i=0$, to $r(x)$ jest poprawnym słowem kodowym.

Syndromy są wyliczane w procesie dekodowania, aby wykryć błędy w odebranym słowie kodowym.

Macierzowy wzór dla syndromów:

$S=H\cdot r^T$

gdzie:

- *S* – wektor syndromów.
- *H* – macierz kontroli parzystości.
- r^T – transponowany wektor odebranego ciągu.

5. Algorytm Berlekampa-Masseya (lokalizacja błędów)

Wzór:

Wielomian lokalizatora błędów $\sigma(x)$ spełnia równanie:

$\sigma(x) \cdot S(x) \equiv 0 \pmod{x^{2t}}$

gdzie $S(x)$ to wielomian syndromów:

$S(x) = S_1 + S_2 x + S_3 x^2 + \cdots + S_{2t} x^{2t-1}$.

Opis:

Ten wzór jest używany w algorytmie Berlekampa-Masseya do znalezienia wielomianu $\sigma(x)$, który wskazuje pozycje błędów.

Proces dekodowania wykorzystuje obliczone syndromy i algorytmy, takie jak Berlekampa-Masseya, aby zlokalizować i poprawić błędy.

6. Algorytm Chiena (sprawdzanie pierwiastków)

Wzór:

Dla każdego $x=\alpha^j$, $j=0,1,\dots,n-$, oblicza się:

$\sigma(\alpha j)=1+\sigma_1\alpha^j+\sigma_2(\alpha^j)^2+\cdots+\sigma_t(\alpha^j)^t$.

Jeśli $\sigma(\alpha^j)=$, to pozycja *j* zawiera błąd.

Opis:

Algorytm Chiena służy do znajdowania pozycji błędów na podstawie wielomianu lokalizatora błędów $\sigma(x)$.

7. Obliczanie wielkości błędów (wzór Forneya)

Wzór:

Wielkość błędu e_j na pozycji *j* oblicza się jako:

$e_j=\frac{\Omega(\alpha-j)}{\sigma\prime(\alpha-j)}$

gdzie:

- $\Omega(x)$ to wielomian błędu, wyznaczany jako: $\Omega(x)=S(x)\cdot\sigma(x) \pmod{x^{2t}}$.
- $\sigma'(x)$ to pochodna wielomianu $\sigma(x)$:

  $\sigma'(x)=\frac{d}{dx}\sigma(x)$.

Opis:

Ten wzór pozwala obliczyć wartości błędów, które należy poprawić w słowie kodowym.

8. Korekcja błędów

Wzór:

Poprawiony ciąg $r'(x)$ oblicza się jako:

$r'(x)=r(x)+e(x)$,

$e(x)$ to wielomian błędów, którego współczynniki są zerami z wyjątkiem pozycji błędów.

Opis:

Proces korekcji błędów polega na dodaniu wykrytych błędów do odebranego słowa.

Pełne Kroki Kodera i Dekodera

Koder:

1. Zdefiniowanie parametrów kodu n,k,t.
2. Skonstruowanie wielomianu generatora $g(x)$.
3. Ustalenie macierz generatora G.
4. Zakodowanie wiadomości $u(x)$ za pomocą $c(x)=u(x)\cdot G$.

Dekoder:

1. Odbierz zakodowane słowo $r(x)$.
2. Oblicz syndromy SS za pomocą $S=H\cdot r^T$.

3. Jeśli wszystkie syndromy są zerowe (S=0), brak błędów.
4. Wyznacz wielomian σ(x) (lokalizatora błędów) za pomocą algorytmu Berlekampa-Masseya.
5. Znajdź pozycje błędów za pomocą algorytmu Chiena.
6. Oblicz wielkość błędów e_j za pomocą wzoru Forneya.
7. Skoryguj odebrane słowo:r’(x)=r(x)+e(x)

Przykład kodowania i dekodowania z użyciem kodu BCH o parametrach (7,4,3)(7,4,3), co oznacza:

- n=7n=7: długość kodu,
- k=4k=4: długość wiadomości,
- n−k=3$n-k$=3: liczba bitów korekcji,
- Kod może naprawić t=1t=1 bit błędu (bo d=2t+1=3d=2t+1=3).

1. Wielomian Generujący

Przyjmijmy, że wielomian generujący kod BCH to:

$g(x)=x^3+x^2+1$

2. Macierze *G* i *H*

Macierz generująca *G* i sprawdzająca *H* są wyznaczane na podstawie $g(x)$.

Macierz generująca *G*:

$$G = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix}$$

- Pierwsze cztery kolumny to macierz jednostkowa I_4,
- Kolejne trzy kolumny są wyznaczone na podstawie współczynników $g(x)$.

Macierz sprawdzająca *H*:

$$H = \begin{bmatrix} 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 1 \end{bmatrix}$$

- Transpozycja *P* (z macierzy *G*) i macierz jednostkowa I_3.

3. Kodowanie wiadomości

Załóżmy wiadomość m=[1,0,1,1]

Zakodowana wiadomość *c* to:

$c = m \cdot G$

$$c = [1,0,1,1] \cdot \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix} = [1,0,1,1,0,1,0]$$

Zakodowana wiadomość to c=[1,0,1,1,0,1,0].

4. **Odebrana wiadomość *r* z błędem**

Przyjmijmy, że odebrana wiadomość r*r* różni się od c*c* w jednym bicie:

r=[1,1,1,1,0,1,0]

5. **Obliczanie syndromu – sprawdzenie poprawności**

Obliczamy syndrom:

$s = r \cdot H^T$

$$s = [1,1,1,1,0,1,0] \cdot \begin{bmatrix} 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 1 \end{bmatrix}^T$$

Po obliczeniach:

s=[1,1,0]

Syndrom s≠0, co oznacza błąd.

6. **Lokalizacja błędu**

Syndrom s=[1,1,0]*s*=[1,1,0] wskazuje pozycję błędu. W tym przypadku odpowiada to pozycji 2

7. Korekcja błędu

Poprawiamy odebrany wektor *r*, zmieniając bit na pozycji 2:

r=[1,0,1,1,0,1,0]*r*=[1,0,1,1,0,1,0]

Po korekcji wiadomość *r* zgadza się z zakodowaną wiadomością *c*.