Wstęp

Kod Hamminga to blokowy, liniowy kod korekcyjny wynaleziony przez Richarda Hamminga. Jego zdolność korekcyjna wynosi t = 1, z tego powodu w celu uzyskania niezawodnej transmisji całą wiadmość należy podzielić na słowa (zakodowane kodem Hamminga), które po przejściu przez kanał transmisyjny uzyskują przeważnie przekłamanie tylko jednego bitu – maksymalna oczekiwana odległość Hamminga między słowem transmitowanym a odbieranym równa 1.

Dla każdej liczby całkowitej istnieje kod Hamminga

– liczba pozycji kontrolnych

- długość zakodowanego słowa

- ilość bitów informacji w słowie

Wszystkie takie kody Hamminga mają minimalną odległość Hamminga pomiędzy wektorami kodowymi . Zgodnie ze wzorem na zdolnosć detekcyjną wszystkie wykrywają do 2 bitów przekłamań w słowie, oraz mają zdolność korekcyjną czyli , ponieważ przy jednym przekłamanym bicie, jest możliwość jednoznacznie określenia wiadomości początkowej ale już przy dwóch błędnych bitach, najmiejsza odległość hamminga pomiędzy otrzymaną wiadmością a danym wektorem kodowym będzie wskazywała niepoprawny wektor kodowy.

Na przykład dla m= 3 pozycji kontrolnych otrzymamy kod (7,4) w którym zakodowane słowo będzie zawierać 7 bitów, z których 4 będą wiadomością.

GENERALNY ALGORYTM KODOWANIA

Przyjmijmy, że bity parzystości znajdują się na pozycjach będących potęgami 2. Algorytm jest następujący:

1. wszystkie pozycje będące potęgami 2 (1, 2, 4, 8, 16,...) są bitami parzystości,
2. wszystkie pozycje niebędące potęgami 2 (3, 5, 6, 7, 9, 10,...) to bity informacyjne,
3. każdy bit parzystości wskazuje parzystość pewnej grupy bitów w słowie, a jego pozycja określa, które bity ma sprawdzać, a które opuszczać:

* pozycja 1: opuszcza 0 bitów, sprawdza 1 bit, opuszcza 1 bit, sprawdza 1 bit, opuszcza 1 bit itd. (1, 3, 5, 7, 9,...),
* pozycja 2: opuszcza 1 bit, sprawdza 2 bity, opuszcza 2 bity sprawdza 2 bity, opuszcza 2 bity itd. (2, 3, 6, 7, 10, 11,...),
* pozycja 4: opuszcza 3 bity, sprawdza 4 bity, opuszcza 4 bity sprawdza 4 bity, opuszcza 4 bity itd. (4, 5, 6, 7, 12, 13, 14, 15,...)
* pozycja 8: opuszcza 7 bitów, sprawdza 8 bitów, opuszcza 8 bitów sprawdza 8 bitów, opuszcza 8 bitów itd.,

...

* pozycja n: opuszcza n-1 bitów, sprawdza n bitów, opuszcza n bitów, sprawdza n bitów itd.

Kodowanie Hamminga to poprostu użycie dodatkowych bitów parzystości, które pozwolą skorygować błąd. W 7-bitowej wiadomości jest możliwe 7 przekłamań pojedyńczego bitu, dlatego 3 bity kontrolne pozwalają skorygować > 7 błedów.

Zaprezentujmy przykładowe kodowanie dla kodu Hamminga(15,11)

Bity parzystości są umiejsciowione na pozycjach odpowiadających kolejnym potęgą 2. Kodowanie należy dobrać tak aby bity parzystości nie zależały od siebie. Kodując według podanego algorytmu nieprawidłowe bity parzystości utworzą liczbę (syndrom – wektor) odpowiadającą pozycji przkłamanego bitu. to wiadomość, to bity parzystości dla bitów oznaczonych gwiazdką w danym wierszu.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | **11** | **12** | **13** | **14** | **15** |
|  | \* |  | \* |  | \* |  | \* |  | \* |  | \* |  | \* |  | \* |
|  |  | \* | \* |  |  | \* | \* |  |  | \* | \* |  |  | \* | \* |
|  |  |  |  | \* | \* | \* | \* |  |  |  |  | \* | \* | \* | \* |
|  |  |  |  |  |  |  |  | \* | \* | \* | \* | \* | \* | \* | \* |

dla dla ,13,15

dla dla

Gwiazdki w danym wierszu oznaczają numery bitów dla których jest obliczany dany bit parzystości. Jak widać każdy bit wiadomości ma unikalną kombinacje bitów parzystości którego biorą go pod uwage przy obliczaniu bitu parzystości oraz tych które go nie uwzględniają. Jeżeli na przykład jeden bit na danej pozycji byłby przekłamany to informacja uzyskana z obliczenia w dekoderze które, z 4 ciągów bitów są parzyste(lub nie), jednoznacznie zidentyfikuje miejsce przekłamania. Na przykład dla przekłamania pierwsze 3 ciągi bitów bedą nieparzyste(lub parzysty jeżeli bit parzystości prawidłowo będzie tworzył ciągi nieparzyste), a 4 będzie parzysty(lub nieparzysty ...). Proces kodowania można zapisać jak mnożenie wektora danych z macierzą generacji G wynikającą z powyższej tabeli. . Proces detekcji błedu (sprawdzenia parzystości) jako mnożenie zakodowanego słowa z macierzą , a nastepnie poddaniu jej operacji mod 2 syndrom to wektor wskazujący czy(i gdzie) wystąpił błąd, w przypadku braku przekłamania powinien być wektorem zerowym. Proces korekcji jak korekcje bitu na który wskazuje syndrom (Jeśli kodowanie jest zaprojektowane tak aby syndrom wskazywał pozycje bitu przekłamanego). Proces Dekodowania jako pomnożenie przez macierz R której zadaniem będzie „wyłuskanie” bitów informacji ze słowa. Macierz R będzie miała kolumny równe 0 na pozycji bitów parzystości, a reszta kolumn będzie miała jedną 1 odpowiadającą danej pozycji wiadomości.

Naszym zadaniem projektowym nie jest implementacja koderów i dekoderów dlatego do kodowania oraz dekodowania użyjemy narzędzi dostępnych w Matlabie – „Communications Toolbox”

Symulator

Jest to skrypt (klasa) pozwalający na analizę statystyczną kodów korekcyjnych

Używa, funkcji pomocniczych:

* generateData – służy do generacji wektora danych binarnych
  + Używa funkcji matlabowej randi która generuje liczby pseudolosowe z rozkładem równomiernym
* encodeHamming – służy do zakodowania danych
  + Używa matlabowej funkcji encode zauważyliśmy, że funkcja matlabowa encode używa innego kodowania niż ten podany jak główny algorytm. Bity parzystości w funkcji encode bity parzystości są zawsze na początku słowa zakodowanego
* decodeHamming – służy do odkodowania zakodowanych danych
* Używa matlabowej funkcji decode
* bscChannel – modeluje kanał komunikacyjny - binary symmetric channel (BSC) w którym bity mogą zostać błednie odebrane w zależności od zadanego prawdopodobieństwa
* Używa matlabowej funkcji bsc

Jest to Model Bernulliego który wprowadza pojedyncze, niezależne błedy.

A close up of text on a white background

Description automatically generated

* bncChannel – modeluje kanał komunikacyjny – burst-error channel którym pojawiają się błedy grupowe. Jest to dwustanowy model Gilberta parametryzowany 3 wartościami:
* Prawdopodobieństwo przejścia ze stanu dobrego do złego - = Good2Bad =G2B
* Prawdopodobiestwo przejścia ze stanu złego do dobrego – = Bad2Godd = B2G
* Prawdopodobieństwo wystąpenie błedu w stanie złym - = loss density

A close up of text on a white background

Description automatically generated

Założyliśmy że w stanie dobrym błedy nie występują, aby ułatwić operowanie kanałem z błedami grupowymi parametr r został wyrażony poprzez średnią długość błedu grupowego

ABEL a parameter p poprzez prawdopodobiestwo wystąpienia błedu grupowego dane wzorem

Instrukcja używania symulatora

...

# Źródła

Mochnacki, Władysław. *Kody korekcyjne i kryptografia*. Oficyna Wydawnicza Politechniki Wrocławskiej, 1997. https://www.dbc.wroc.pl/Content/442/mochnacki\_kody.pdf

Wikipedia contributors. (2020, April 7). Hamming(7,4). In *Wikipedia, The Free Encyclopedia*. Retrieved 17:48, April 19, 2020

Maciejewski, H., Jarnicki, J. and Woda, M., 2017. *W11 – Kody Nadmiarowe, Zastosowania W Transmisji Danych*. [online] Zsk.ict.pwr.wroc.pl. Available at: <http://www.zsk.ict.pwr.wroc.pl/zsk/repository/dydaktyka/ndsc/wyklady/niezawodnosc\_w11\_12.pdf> [Accessed 19 April 2020].

Gaussianwaves.com. 2020. *Hamming Codes – How It Works – Gaussianwaves*. [online] Available at: <https://www.gaussianwaves.com/2008/05/hamming-codes-how-it-works/> [Accessed 19 April 2020].

Eduinf.waw.pl. 2020. *Bit W Zastosowaniach - ECC*. [online] Available at: <https://eduinf.waw.pl/inf/alg/002\_struct/0010.php> [Accessed 19 April 2020].

Mathworks.com. 2020. *Uniformly Distributed Pseudorandom Integers - MATLAB Randi*. [online] Available at: <https://www.mathworks.com/help/matlab/ref/randi.html> [Accessed 20 April 2020].

Mathworks.com. 2020. *Error Detection And Correction- MATLAB & Simulink*. [online] Available at: <https://www.mathworks.com/help/comm/error-detection-and-correction.html?s\_tid=CRUX\_lftnav> [Accessed 20 April 2020].

Mathworks.com. 2020. *Binary Symmetric Channel - MATLAB Bsc*. [online] Available at: <https://www.mathworks.com/help/comm/ref/bsc.html> [Accessed 20 April 2020].

Dde.binghamton.edu. 2020. *Binary Additive White-Gaussian-Noise Channel*. [online] Available at: <http://dde.binghamton.edu/filler/mct/lectures/25/mct-lect25-bawgnc.pdf> [Accessed 20 April 2020].

Mathworks.com. 2020. *Channel Models- MATLAB & Simulink*. [online] Available at: <https://www.mathworks.com/help/comm/channel-models.html> [Accessed 20 April 2020].

Dip.ee.uct.ac.za. 2020. *Probability Of Error In Transmission*. [online] Available at: <https://www.dip.ee.uct.ac.za/~nicolls/lectures/eee482f/05\_proberr.pdf?fbclid=IwAR0jN40ugcdV\_BYJ1GXyBhoWcbj4ODz0mVEFp6-xbV\_Vh0PeVfCFi4phtL8> [Accessed 20 April 2020].